uc3m | Universidad Carlos III de Madrid

Grado en Ingeniería en Tecnologías de Telecomunicación
2016/2017

Trabajo Fin de Grado

Soporte de NFC en Wifi-Direct

José Esteban-Infantes Ruiz

Tutor/es
Andrés Marín López

Leganés 9 de Octubre de 2017

Agradecimientos

Quiero agradecer a mi familia, en especial a mis padres por su esfuerzo y paciencia durante todos estos años.

A mi familia del Budo, por ayudarme con esa magia y felicidad desde el tatami.

A todos mis amigos y novia, que no han tenido más remedio que aguantar mis quejas y no siempre entre cervezas.

Por último quiero dedicar este trabajo a mi abuelo Carlos, persona a la que siempre he admirado y desde joven me animó a estudiar esta carrera.

Por las penas que anegaron el camino,
el sacrificio
y las noches sin dormir,
por las risas que nos dieron,
y las risas por venir.

Gracias a todos.

Abstract

WiFi-Direct is a technology capable of connecting different types of devices, providing a wide range of flexibility and interconectivity; with the help of the wpa_supplicant we can also manage this type of connections for all Linux systems.

Another widely used technology present in many devices like mobile phones is NFC, which allow us to establish a small communication between devices by bringing them very close to each other; this communication is enough to bootstrap a more capable wireless connection.

The aim of this project is to study the support of NFC in a WiFi-Direct connection. We will take advantage of the wpa_supplicant in Linux to establish the Wi-Fi Direct connection between a PC and a mobile phone, using the NFC capabilities, achieving a more secure environment to launch this type of connections.

Contents

List of Figures

List of Tables

1 Introduction

1.1 Motivation

NFC technology presents a great opportunity to interact and establish connections between devices. Although this connections are small they are very resource-full, allowing us to exchange small files and simple information by simply bringing both devices close to each other (about 4cm). This feature seems a little odd since we are speaking about communications, but it is indeed a very powerful security measure, for the user will have to be in a very close range for the technology to work. In a case scenario in which an attacker would want to steal some information stored in a NFC-TAG, the attacker would have to physically approach to the victim within a 4cm distance with a device capable of reading this information. NFC has many possibilities once there has been "contact", but for us the main advantage of NFC will be the capability to launch a more complex connection. This means we can use a NFC-TAG to start a more suitable connection; a very common example of this use would be the contact-less credit card payment, which just by bringing our credit card close to the terminal, and a connection to make the payment is established. NFC operates in the high frequency band of 13,56 MHz, which is globally available and presents no trouble when in regard of this close-up-range technology.

Nowadays most smart phones come with NFC capabilities, allowing developers to expand our ways of using this technology; we can see from social networking to secure identification, from commerce to the gaming industry in the last years. But as mentioned before, one of its major advantage and the one we will focus on, is bootstrapping a more capable wireless connection.

There are many possible and very interesting technologies that we could use to establish a more or less complex connection; we could start a file transfer via Bluetooth after reading some NFC-TAG, or simply pair our headset upon touch, the same idea can be applied to set up a Wi-Fi network. Therefore we need to talk about Wi-Fi Direct, a technology that allow us to interconnect different devices without any router intervention, but with the capabilities of Wi-fi connectivity. Bringing flexibility to the traditional way of communication, for we can establish a connection between two devices (peer to peer)or more (group P2P), reaching a distance up to 250m with a maximum transmission rate of 250 Mbps, which makes this a technology worth knowing.

Wi-Fi-Direct can be found in devices like printers, Smart-TVs, tablets, mobile phones, in general all kinds of android devices and even our PCs, most of them found at home, in the office or even the industry, giving a new meaning to the uprising Internet of Things (IoTs) where the concepts of interconnection and independence brought by Wifi-Direct are very clear. But coming back to the PC, we ought to speak about Linux systems and therefore we need to talk about the *wpa_supplicant*; "supplicant" refers to either the hardware or software that is in charge of submitting the credentials to connect the computer to a secure network. The *wpa_supplicant* for Linux, is a software supplicant able to support not only

the WEP and WPA/WPA2 systems but also WPS, allowing a more simple, yet secure set up for our network. WPS in the *wpa_supplicant* will also bring us some interesting options like *WPS_NFC*, of which we are going to take advantage.

On the other hand we are going to use the *wpa_supplicant* in order to build a Wifi-Direct connection. For this purpose we will use another tool of the same family, the *wpa_cli*, that is going to help us not only establish the Wifi-Direct connection but also manage it through commands. For the aim of this project we are taking advantage of the *wpa_cli* in order to establish a Wifi-Direct connection between a mobile phone and a PC.

Regarding the NFC technology, we will use a NFC-Reader device and some NFC-TAGs along with our mobile phone, to study this system and how it performs in Linux. Using the open scripts from nfc-tools we can manage to read or write the NFC-TAGs to fully comprehend how NFC works. With this in mind and by using the capabilities of the previously named *wpa_supplicant*, we will study the possibility of bootstraping a Wifi-Direct connection by using NFC technology.

With this project we intend to bring together two very powerful technologies that have been present in our daily basis for a few years now, but still are not very known even though we have it in our pockets. We seek to break this barrier and open a wide range of opportunities for developers, so that in the future we can enjoy the capabilities of both Wifi-Direct and NFC in a unique manner.

1.2 Objectives

- **General Objective:**

  The main objective of this project will be studying the capabilities of NFC in order to establish a Wifi-Direct connection between a PC and a mobile phone, thus bringing two very different technologies together in order to open a wide range of possibilities for the future of communication among different devices, as well as its security.

- **Specific Objective:**

  1. Study of the Wi-Fi Direct[1] technology.
  2. Study of the *wpa_supplicant* and *wpa_cli* for a better understanding of its capabilities.
  3. Study of the NFC technology.
  4. Determine the use of NFC in Linux with nfc-tools.
  5. Show and test the use of a NFC-TAG to establish a simple P2P connection.
  6. Show and test the use of a NFC-TAG in a more complex P2P connection.
  7. Create a simple manual recovering the steps taken to create P2P connection with the use of NFC

1.3 Memory Content

In the second chapter, the Estate of the Art, we will see a further description of the technologies appointed for this project, taking a more detailed but general look of how Wifi-Direct and NFC work.

The third chapter will be about *wpa_supplicant* and Wifi-Direct, but focused on the aspects needed for our project. We will see its architecture, connection modes, *wpa_cli*, wps, and so on. All aimed to WPS_NFC.

In the fourth chapter we will focus on NFC technology, a brief history, its architecture, nfc-tools for linux, reading and writing NFC-TAGS.

The fifth chapter will explain the proposal to bring this technologies together and dedicate to the tests.

The sixth chapter will include a brief summary of the project, as well as the conclusions and discuss the future lines of work.

1.4 Technology Used

- Computer Dell Studio XPS 8100, 8Gb RAM, Processor Intel i7. Windows 10 64 bits
- VirtualBox with O.S Debian 8
- Mobile phone HUAWEI G7-L01 with Android version 5.1.1 and Kernel 3.10.49
- 2 Wireless Adapter TP-LINK TL-WN722N
  - Standard IEEE 802.11/b/g/n
  - Frequency range: 2,4-2,4835 GHz
  - Transmission speed: 150Mbps
  - Security WEP,WPA/WPA2,WPA-PSK/WPA2-PSK
  - Modes Ad-Hoc and Infrastructure
  - Supports WPS
- Contactless Smart Card Reader SCL3711
  - Supports ISO14443 Type A and B
  - Supports MIFARE™, FeliCa™ and NFC forum tag
  - Support ISO/IEC 18092 peer-to-peer protocol
  - Frequency band of 13.56 MHz
  - Baud Rate (to Transponder): 848 Kbps
- 2 MIFARE Ultralight EV1 Smartcard
- FeliCa Lite Smartcard

2 Estado del Arte

Siempre se ha entendido que el propósito de la tecnología consiste en facilitar la vida de las personas, para que las tareas más complicadas o irrealizables lo dejen de ser, y todas aquellas acciones consideradas cotidianas pasen a ser un mero recuerdo. De esta forma el ser humano ha podido perdurar y progresar, siendo capaz de realizar un sinfín de tareas otrora impensables en el menor tiempo posible; evolución que ha supuesto una verdadera monetización del tiempo, tiempo que hoy por hoy procuramos malgastar lo menos posible. Muchas tecnologías han visto la luz gracias a este pensamiento, con la idea de facilitar la intercomunicación de las personas en el menor tiempo posible, con el menor desplazamiento posible.
Cuando hablamos de comunicación, lo primero que se nos viene a la cabeza es Internet, es la cualidad de poder compartir información en casi tiempo real a cualquier parte del mundo: nuestras fotos, música, vídeos, documentos, películas y demás. Pero de alguna forma esta tecnología siempre ha estado limitada en un entorno dependiente. Si hablamos de comunicación entre ordenadores a través de internet, es evidente que la participación de un router es inevitable ya sólo por mero hecho del acceso a la red. ¿Pero por qué ha de ser así en otros contextos? Imaginemos un entorno de oficina o la disposición de nuestro hogar, situaciones cotidianas en las que podemos encontrar uno o más PCs conectados a la misma red, así como impresoras y otros dispositivos, donde aparece una necesidad real de comunicación entre estos dispositivos.

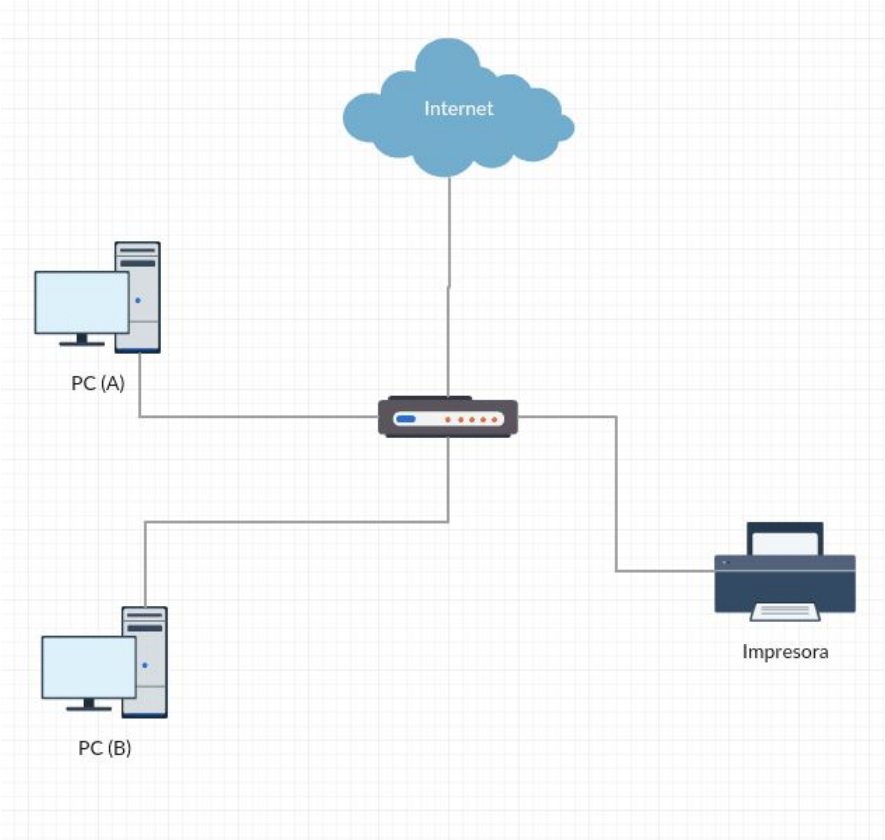

Figura 1: Small Office/Home Network

En este entorno que nos presentan las redes clásicas[1], podemos observar que todas las comunicaciones ocurren a través del router, obligando en este caso a que exista una conexión física mediante cable ethernet. Y que en cualquiera de los casos nunca podríamos hablar de comunicación directa entre los dispositivos, puesto que toda la información ha de pasar por el router, lo que nos presenta no solo un problema a la hora de establecer las comunicaciones, sino que ya estaríamos hablando de problemas de seguridad, puesto que cualquier atacante podría interceptar nuestras comunicaciones.
A día de hoy las redes presentes en el hogar y pequeñas oficinas, no han cambiado demasiado. Aparece la tecnología Wi-Fi que nos facilita la conexión con el punto de acceso, en cuanto a versatilidad y flexibilidad. Ya no dependemos del cable para establecer nuestras comunicaciones y eso posibilita la conexión de un mayor número de dispositivos de diferentes, como smart phones o tabletas entre otros.

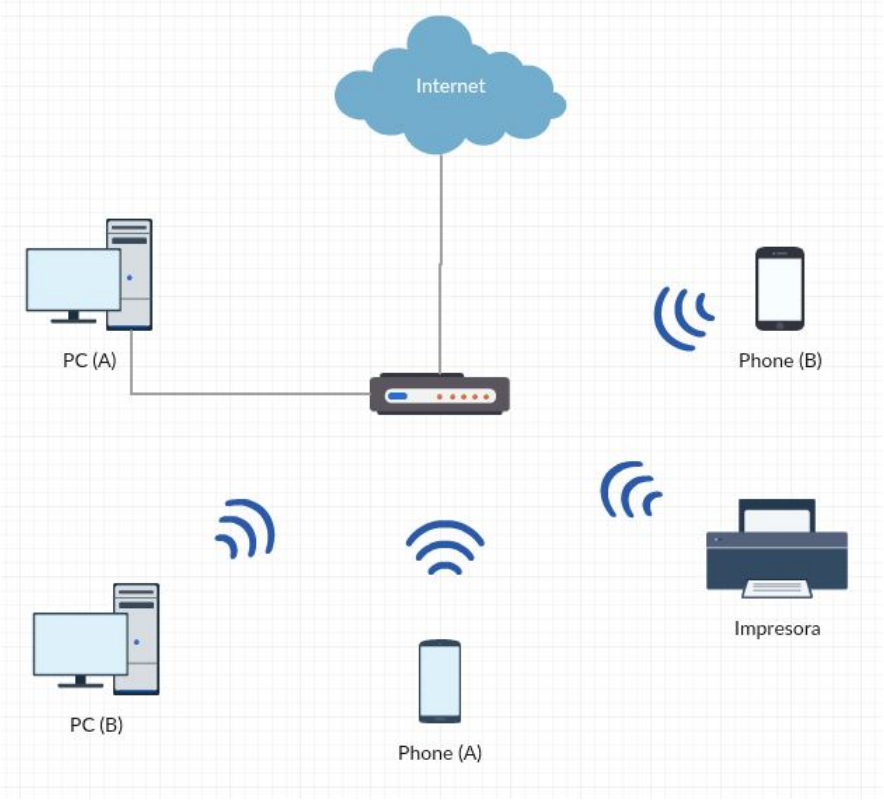

Figura 2: Wi-Fi Network

No obstante en esta nueva red Wi-Fi[2] continúan los problemas planteados anteriormente. Seguimos sin ser capaces de establecer una comunicación directa entre los dispositivos, lo cual no nos proporciona una solución a nuestro problema, dependemos del rotuer para poder llevar a cabo nuestras conexiones. Es más, los problemas de seguridad incrementan, puesto que cualquier atacante podría interceptar nuestras comunicaciones y conectarse a la red local sin necesidad de un punto de acceso físico.

Nuevamente damos un salto tecnológico y aparece una solución a nuestro pro-

blema, el por todos conocido Bluetooth. Permitiendo una comunicación entre dispositivos bastante sencilla y fácil de utilizar, con velocidades de hasta 25 Mbps en la versión cuatro. De esta forma ya podríamos montar nuestra red con dispositivos interconectados entre sí uno a uno, como podemos tener el móvil y los auriculares Bluetooth, compartir archivos entre dos móviles o incluso entre dos ordenadores que cuenten con esta tecnología, siempre y cuando estos se encuentren en un área de no más de 10 m. Esta tecnología aporta un nivel de flexibilidad e interconexión muy superior al que habíamos alcanzado en las redes Wi-Fi[2], podemos conectar los dispositivos Bluetooth entre sí sin necesidad de un router en en el centro de las conexiones, simplemente los dos dispositivos.
Sin embargo llega un momento que necesitamos una tecnología con mayor capacidad, que nos permita conectar dispositivos no compatibles de una forma rápida, fácil y segura. Aquí aparece la segunda solución sobre la que vamos a trabajar, Wi-Fi Direct[1]. La idea es simple, como podemos imaginar es muy similar a Bluetooth pero con las ventajas que nos proporciona el Wi-Fi. Esta tecnología supone un cambio importante en la forma de entender las redes más clásicas[1][2] vistas anteriormente, y sobrepasando las limitaciones que se nos habían presentado en nuestra primera solución. Con Wi-Fi Direct podemos establecer conexiones entre diversos dispositivos alcanzando tasas de 250 Mbps en un radio equivalente al del Wi-Fi (250 m). Este tipo de comunicación no solo expande el marco de oportunidades para aplicaciones de retransmisión de vídeo como es el caso de Miracast. También ofrece una comunicación segura e individual para grupos de dispositivos, logrando una versatilidad imposible de conseguir en las redes anteriores [1][2]. A continuación mostramos un ejemplo de una red simple en la que se emplea Wi-Fi Direct.

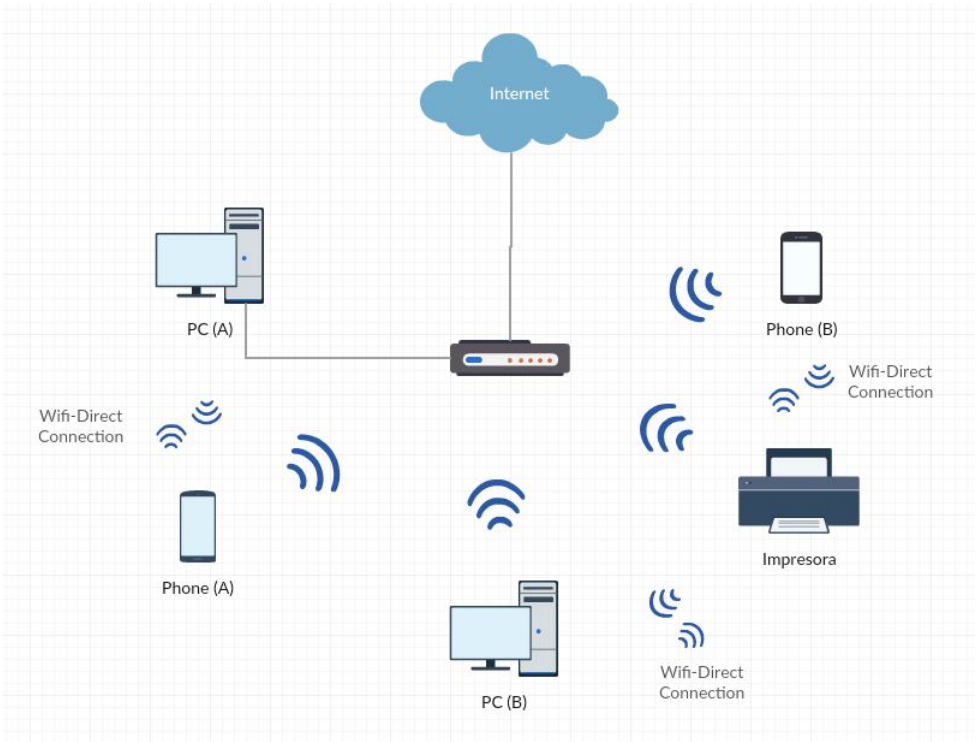

Figura 3: Wi-Fi Direct Network

Como podemos apreciar en la imagen[3], esta nueva red, a diferencia de las redes clásicas[1][2], presenta una serie de dispositivos interconectados entre sí mediante Wi-Fi Direct y al mismo tiempo conectados a su punto de acceso de formas diferentes. Se puede observar la comunicación de un PC(A) y un smart phone(A), así como de otro PC(B) y otro smart phone(B) con una impresora. Todas estas conexiones pueden ser independientes (PC(A) y Smart Phone(A)) o en grupo, como ocurre con la impresora; en cualquier caso todos estos dispositivos comparten y transmiten información sin la necesidad de un router para establecer la comunicación.
Cabe añadir la mayor seguridad[3] que presenta Wi-Fi Direct frente a su competidor, haciendo de esta una tecnología muy interesante de cara al futuro. Ya sea en las redes de oficinas, como en el hogar, y aún más apetecible para la industria del Internet de las Cosas (Internet of Things) y la domótica, facilitando comunicación de todos los dispositivos del hogar entre sí, para luego ser controlados por un smart phone o una tableta, por ejemplo. Redes que antes podían resultar impensables sin algún tipo de unión para estos dispositivos.

Sin embargo este enfoque en las comunicaciones resulta demasiado alienado, de alguna forma la propia palabra nos evoca esa sensación de distancia. Incluso cuando hablamos de comunicación entre dos personas, no pensamos en una simple conversación cara a cara. Este pretexto lo hemos planteado cuando buscábamos una alternativa a nuestras redes clásicas [1][2], pero no hemos terminado de cerrar la situación. Si planteamos la situación de dos personas conversando entre ellas, tal vez sí nos valga nuestra solución y nos sirvan nuestras "redes Wifi-Direct[3]". Pero nuevamente estamos cayendo en el mero hecho de la conversación, cuando las personas así como la tecnología en este ámbito, buscan la comunicación o en otras palabras, el intercambio de información. Por supuesto esta información puede tratarse de cualquier cosa, incluso cuando lo extrapolamos al ejemplo de las dos personas, esta información puede ser únicamente la conversación en sí mismo; pero todos tenemos claro que esa información también puede consistir en compartir un archivo de información, o por lo que a las dos personas corresponde, un libro, una película, un documento. Todos estos elementos se pueden compartir o prestar como una forma de comunicación; si nuestro receptor estuviese muy lejos, podríamos emplear sistemas de mensajería externos para hacerle llegar el elemento en cuestión. Pero esto no es así cuando se trata de personas que se encuentran físicamente cerca de nosotros, dada esta situación nos acercamos a la persona y le cedemos el objeto que queramos; existe una cercanía física y un casi contacto para poder entablar esa comunicación y poder realizar el intercambio en cuestión.
Esto nos conduce a preguntarnos si podría existir una tecnología que funcionase bajo esta premisa, buscando la cercanía física para establecer la comunicación en lugar de la larga distancia, la respuesta es sí. Pero para ello debemos cambiar la aproximación de nuestra solución y remontarnos a una tecnología más que extendida. "RFID", siglas de "Radio Frequency Identification" o 'Identificación por Radio Frecuencia'. El propósito de este sistema es transmitir el identificador de un objeto mediante radio frecuencia, para lo que se emplean unas pequeñas antenas (que incluso se adhieren a través de pegatinas) y son capaces de establecer una

comunicación por radio frecuencia en distancias de hasta 100m, empleando las bandas de 125 KHz y 134.2 KHz en baja frecuencia, 13.56 MHz en alta frecuencia. Existen también dispositivos en bandas de frecuencia mucho más elevadas, conocidos como "UHF", "Ultra High Frequency" que emplean el rango de 868MHz a 956MHz, e incluso microondas a 2.45 GHz en las que la tecnología logra el mayor alcance;de esta forma aplicamos una clasificación a los sistemas RFID en función de las bandas que empleen. Cabe decir que los dispositivos que funcionan en UHF no están regulados de forma internacional, por lo que no los veremos; mientras que los de baja frecuencia pueden emplearse libremente.
Esta tecnología presenta una alternativa para la comunicación entre dispositivos, que ha facilitado en gran medida la forma de trabajar en ciertos sectores de la industria. Un ejemplo muy claro se puede ver en las cadenas de súper producción donde pueden tener controlados su manufacturación a través de pegatinas RFID, la seguridad de una tienda en la que los productos lleven pegatinas, o incluso el acceso en oficinas y lugares de acceso limitado.

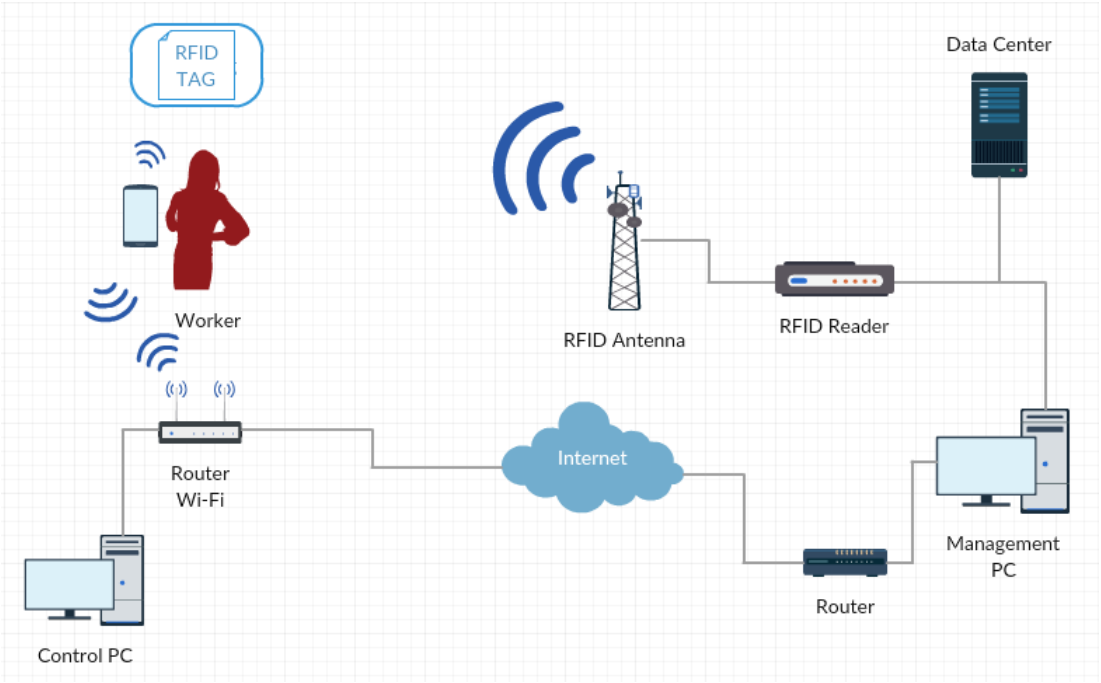

Figura 4: RFID Network

La disposición que aparece en la imagen[4] podría ser una red más o menos compleja con sistema RFID integrado. Podemos continuar con el ejemplo del comercio y el almacén. Un producto con un TAG-RFID en forma de pegatina es registrado en el almacén del distribuidor, la pegatina es detectada e identificada por una antena RFID y leída por su correspondiente dispositivo lector, la información es procesada en un ordenador encargado de esta tarea, en nuestro caso el "Management PC". Los datos sobre el producto se almacenan en un "Data Center" para un futuro. Entonces el producto es distribuido a una tienda, en la que un trabajador podrá emplear una máquina lectora de RFID que se comunique con el ordenador de la tienda, para sí registrar los productos que le van llegando del almacén. Por último podríamos emplear esta información para cotejarla con el al-

macén a través de Internet y comprobar así que la mercancía recibida es correcta, o simplemente comunicar el stock restante de algún producto. Sin embargo esta última parte ya entraría dentro de las aplicaciones que pueden llevarse a cabo con el soporte de RFID.
También es importante pararse un poco y hacer el análisis de esta red, específicamente en las partes que envuelven RFID; vemos que aparecen dos posibilidades de comunicación física más lejana en función de las capacidades del sistema que estemos empleando. Esto ya supone una solución al problema que planteábamos inicialmente cuando hablábamos de las comunicaciones entre dos dispositivos de forma directa. Pero debemos tener en cuenta que esta comunicación es unidireccional, puesto que solo se produce contra las TAG-RFID en las que o bien escribimos, o bien leemos la información contenida, nada más. Pero también presenta un inconveniente de seguridad, puesto que un atacante equipado con un lector RFID podría ser capaz de leer la información de nuestros TAG-RFID a distancias de hasta 100 metros. Tal vez esta situación no resulte muy fácil de imaginar en un entorno industrial como los tratados hasta el momento ¿pero qué ocurre si lo trasladamos al hogar o la oficina? Al igual que ocurría con las comunicaciones Wireless, no queremos que nuestro vecino o compañero fisgue en la información de nuestras TAG-RFID.
Por tanto podemos afirmar que esta solución no nos resulta válida ya no por la seguridad, sino por el simple hecho de que no podemos intercambiar información entre dos dispositivos, siempre hay que hacerlo por medio de una TAG-RFID lo cual nos limita a alcanzar nuestro objetivo de intercambio de archivos e información de forma directa entre dispositivos.

Sin embargo podríamos modificar esta tecnología de forma que funcionase en un rango de frecuencia específica, asegurando que se necesitase una proximidad física mucho mayor, de forma que fuera de esa cercanía no funcionase. Pero también necesitaríamos que los dispositivos pudieran comunicarse entre ellos directamente, sin depender de un TAG, pero que al mismo tiempo pudiera emplearse. Con estos requisitos tendríamos una solución mucho más aceptable para el problema que nos acaece.

NFC son las siglas de 'Near Field Communication', 'Comunicación de Campo Cercano'. Esta tecnología nace de la mano de las compañías Nokia, Philips y Sony en el año 2004 con fundación de lo que se conoce como 'NFC forum', que a día de hoy cuenta con más de 170 participantes. El estándar NFC está basado en la implementación de la tecnología RFID incluyendo el estándar ISO/IEC 14443 que definen las características de los dispositivos y las tarjetas de proximidad: forma física, frecuencias de trabajo, señales y protocolos de transmisión entre otras cosas. NFC funciona en la banda de 13.56 MHz, lo que venía siendo la alta frecuencia de RFID, en la que se puede trabajar sin licencia de forma global. Esto hace de NFC una tecnología lo suficientemente general como para que merezca la pena nuestro interés. La idea es sencilla, al acercar dos dispositivos con capacidad NFC muy próximos el uno del otro (hasta los 10 cm), podemos iniciar un intercambio de información:imágenes, vídeos cortos, archivos, documentos o simplemente información en tasas que van desde los 106 kbps hasta los 424 kbps. Al mismo

tiempo NFC también nos permite escribir o leer información de TAG-NFC que recuerdan a los TAG-RFID que empleaba RFID. No obstante los TAG-NFC son variados, existen dos tecnologías principales MiFare que sigue tres de las cuatro partes la norma ISO/IEC 14443, y FeliCa de Sony Japón, que fue rechazado de la norma ISO/IEC 14443. Ambas tecnologías cuentan con un sistema de encriptación y protección para las tarjetas. Las más populares son las MiFare comercializadas en todo el mundo, mientras que las FeliCa consideradas como RFID smart card, vieron su popularidad en China, Indonesia, Tailandia, Hong Kong, pero en japón mayoritariamente, estando integrada en los sistemas de pago, transporte y demás prestaciones.

Desde el año 2010 son ya muchos los dispositivos smart phone que presentan la tecnología NFC dentro de su repertorio, abriendo un abanico de posibilidades para el desarrollo de aplicaciones basadas en este sistema. Una de las aplicaciones que podemos encontrar, es el intercambio de la información de contacto entre dos terminales.

Figura 5: NFC Information Exchange

Como podemos observar simplemente con acercar los dos dispositivos se produce el intercambio de información. Por supuesto aquí se trata únicamente del ejemplo del intercambio de información de usuario, pero la misma situación sería si quisiéramos transmitir una fotografía o un archivo en general. Lo que efectivamente sirve de solución a nuestro problema, ya que somos capaces de intercambiar información a través de una comunicación directa entre dos dispositivos. Y no sólo eso, sino que existe una seguridad clara por la proximidad física en la que se tiene que establecer la comunicación.
Otra situación muy similar a la de la figura[5], sería cuando uno de los dos dispositivos es en realidad un TAG-NFC, en cuyo caso podríamos encontrarnos con la siguiente situación.

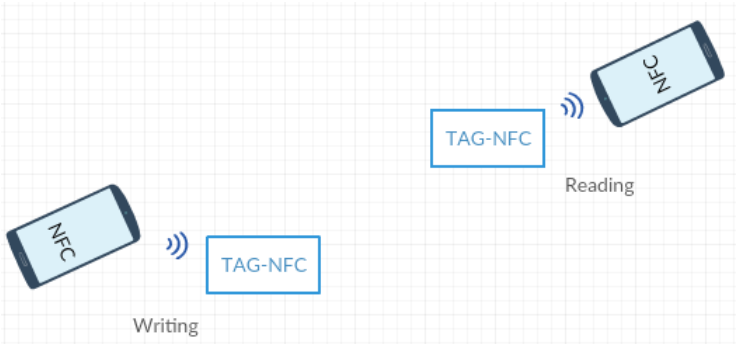

Figura 6: NFC-TAG Writing and Reading

Vemos cómo el mismo TAG-NFC puede ser escrito con información o archivos, que más tarde podemos recuperar desde el mismo dispositivo u otro muy diferente. Esta configuración quizás no resulte tan común pero también se puede dar. Hay que tener en cuenta en esta situación que las NFC-TAG cuentan con capacidad de encriptación y seguridad, por lo que la información vulnerable está a salvo.

Otra de las aplicaciones más habituales en el uso diario es el Contactless Payment, o Pago sin Contacto. Como su propio nombre indica, consiste en efectuar un pago, sin necesidad de introducir la tarjeta de crédito, simplemente acercándola al dispositivo lector, lo que acelera las transacciones sin perder la seguridad propia de la tarjeta bancaria. Lo más interesante es que con una aplicación específica que contuviese los datos bancarios, podríamos realizar el pago simplemente con el móvil, ya que este haría las veces del NFC-TAG que puede ser nuestra tarjeta de crédito.

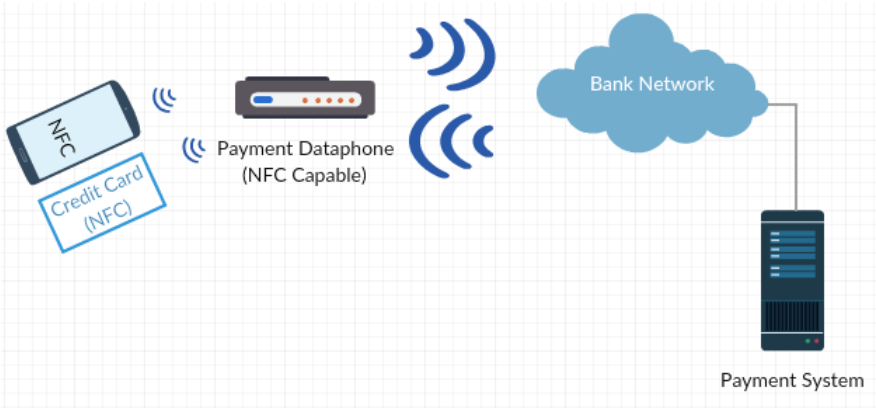

Figura 7: Contactless Payment using NFC

En la imagen[7] se observa cómo el datáfono es capaz de leer la información NFC almacenada en la tarjeta de crédito o bien a través de una aplicación específica del móvil, para realizar el pago sin necesidad de introducir la tarjeta de crédito. Simplemente aproximando alguno de los dos al dispositivo lector; una vez establecida la comunicación y realizado el intercambio de información, el datáfono se conectará a la red del banco en cuestión donde se ejecutarán una serie de procesos para que el pago se lleve a cabo.
Si analizamos con cuidado la situación anterior, figura[7]. Nos daremos cuenta que el pago en sí mismo no se realiza gracias a la tecnología NFC, esta simplemente inicia un proceso. Cuando acercamos la tarjeta o el dispositivo smart phone al datáfono, se desencadena una conexión mayor con los servidores del banco. De alguna forma esto nos hace pensar en por qué no utilizar NFC para establecer conexiones de mayor capacidad. Al fin y al cabo a la hora de generar un intercambio de archivos, NFC está limitado en capacidad. Entonces por qué no buscar la forma de combinar la seguridad y comodidad que nos brinda la tecnología con otra de mayor capacidad. Es cierto que existe la posibilidad de aprovechar Bluetooth con NFC. Pero al principio cuando hablábamos de posibles soluciones para el problema de establecer comunicaciones directas entre dos dispositivos, vimos que Bluetooth no era suficiente y apareció Wi-fi Direct.

Podríamos emplear la seguridad física que brinda NFC con las capacidades de Wifi-Direct para establecer comunicaciones. De esta forma juntaríamos las dos soluciones que existen en nuestro problema: estaríamos estableciendo comunicaciones de forma directa entre dos dispositivos, sin la necesidad de un router en medio de la conexión; y fomentando un establecimiento de esa conexión más seguro y cómodo. Abriendo un campo de posibilidades flexibles y muy variadas a todos los sectores de la industria y servicios. No sólo por las capacidades de las tecnologías en sí, sino por las aplicaciones que pueden desarrollarse. Al fin y al cabo estamos hablando de que tanto Wi-FiDirect como NFC están presentes hoy en día en casi todos los dispositivos smart phone. Con el potencial de ambas unido, se abriría camino a una revolución en la forma de ver las redes y la seguridad, alcanzando situaciones como la siguiente. Donde encontramos un dispositivo smart

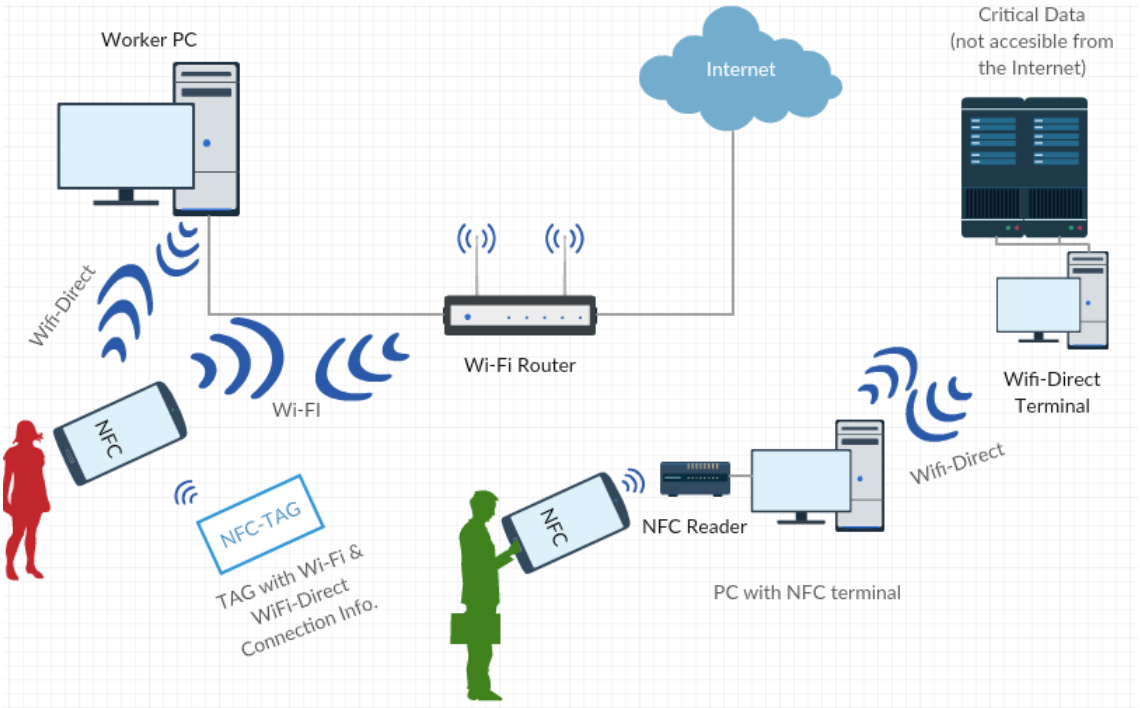

Figura 8: Wi-Fi Direct Network with NFC support

phone que se conecta a la red Wi-Fi y establece una conexión Wi-Fi Direct, a través del 'contacto' con un TAG-NFC que contiene información de ambas conexiones. Por otro lado vemos cómo un usuario accede desde un terminal específico a información sensible no accesible desde Internet, a través de una conexión Wi-Fi Direct lanzada tras la comunicación de su dispositivo smart phone con la información NFC pertinente.

Esta es sólo una pequeña posibilidad de todas las que se abren logrando el soporte de NFC en las conexiones Wifi-Direct. Por lo que a continuación hablaremos de estas mismas tecnologías en el sistema Linux, con el que vamos a trabajar debido a la libertad y control que nos permite.

2.1 Wifi-Direct en Linux

Cuando hablamos de la tecnología Wifi-Direct en PC, encontramos que ya existen integraciones y software para otros sistemas operativos, este no es el caso de Linux. Sistema para el cual tendremos que emplear varias herramientas de software muy diferentes si queremos establecer una conexión Wifi-Direct entre dos dispositivos tan diferentes como un smart phone y un sistema Linux.

En primer lugar hay que destacar cuál será la herramienta fundamental con la que estableceremos nuestras conexiones, *wpa_supplicant*, que también conformará el pilar central de las pruebas prácticas.

Wpa_supplicant, se trata de un programa 'suplicante' encargado de el establecimiento seguro de conexiones inalámbricas, implementando la norma IEEE 802.11/ RSN (Robust Secure Network), por lo que soporta la negociación de claves WEP, WPA y WPA2, convirtiéndolo en componente IEEE 802.1X/WPA de nuestro sistema, funcionando habitualmente como servicio "demonio" (daemon) en segundo plano.

Otra de las capacidades de *wpa_supplicant*, es el soporte de WPS, siglas de "Wi-Fi Protected Setup", mecanismo creado con la idea de facilitar la creación de redes, así como las conexiones de los dispositivos a estas, evitando el uso de claves y contraseñas para establecer la conexión, simplemente con el intercambio de unas credenciales con estos cuatro métodos: PIN, PBC, NFC y USB. De los cuales los dos últimos no se contemplan en el estándar; no obstante *wpa_supplicant* implementa los tres primeros y de los cuales nos vamos a aprovechar.

Wpa_supplicant cuenta con dos herramientas para su manejo: *wpa_cli* y *wpa_gui*. Que consisten en una interfaz de línea de comandos y una pequeña interfaz gráfica respectivamente, capaces de conectarse automáticamente a la primera interfaz inalámbrica controlada por el *wpa_supplicant*, o bien a una especificada por el usuario. Nosotros emplearemos únicamente *wpa_cli* debido a su potencial en el uso de comandos y más concretamente para el manejo de las capacidades Wifi-Direct y WPS_NFC. En el anexo[A] se dejará una lista con los comandos de control necesarios para la práctica.

```
debian@debian:~$ sudo wpa_cli
wpa_cli v2.6
Copyright (c) 2004-2016, Jouni Malinen <j@w1.fi> and contributors

This software may be distributed under the terms of the BSD license.
See README for more details.


Selected interface 'wlan0'

Interactive mode
```

Figura 9: Prompt *wpa_cli*

Sin embargo aún son necesarios algunos componentes más para poder establecer nuestra conexión Wifi-Direct. Véase la necesidad de el 'hostapd, otro software que actúa como 'demonio', permitiendo configurar nuestro sistema Linux como un punto de acceso. Todas las características del mismo, han de ser especificadas en un fichero de configuración. De igual manera habrá que contar con un servidor DHCP para asignar las direcciones IP, máscara de red y demás parámetros al resto de los miembros del grupo P2P cuando establezcamos nuestra comunicación Wifi-Direct.

Por último tenemos que mentar el D-Bus, un sistema de mensajería e intercambio de notificaciones que facilita la comunicación entre procesos con el fin de comunicar aplicaciones entre sí. Esto por un lado, facilita la integración de aplicaciones dentro un mismo entorno, así como un control del correcto ciclo de vida de los procesos relativos a ello; por otro lado controla la comunicación entre procesos del sistema, incluyendo al núcleo además de procesos 'demonio', y la sesión de escritorio.

Con todos estos elementos seremos capaces de establecer una conexión Wifi-Direct entre un terminal Linux y cualquier otro dispositivo con capacidades Wifi-Direct: smart phones, tablets, televisores, otros ordenadores, etc. No obstante, no es el propósito de este proyecto explicar cómo funciona Wifi-Direct ni cómo establecer las conexiones, más allá de lo necesario para poder estudiar el soporte de NFC para esta tecnología, que como ya hemos visto, controlaremos usando el *wpa_supplicant* a través de su *wpa_cli*, empleando los comandos pertinentes para lograr usar "WPS_NFC".

2.2 NFC en Linux

Al igual que nos ocurría con Wifi-Direct, pasa en NFC pero en una menor medida. En otros sistemas operativos algunos fabricantes presentas su propia integración de software para la tecnología, afortunadamente en Linux se cuenta con una comunidad de desarrolladores y voluntarios muy amplia que hace algunos años abrieron el portal de "nfc-tools". Este portal nace con el objetivo de proveer y fomentar el uso de software libre para herramientas del tipo RFID/NFC, ya que en los inicios este mercado estaba completamente limitado por los fabricantes tanto en hardware como en software. Aquí encontraremos las principales herramientas que necesitaremos para dar soporte a nuestro dispositivo NFC y utilizarlo con nuestras TAG-NFC.

La piedra angular del soporte de NFC en Linux es "libnfc", la primera plataforma libre que provee un kit de desarrollo de software para NFC de bajo nivel, junto con una API para programadores. Es estable desde hace tiempo y ha supuesto un gran avance a la hora de desarrollar software con integración de NFC, ahorrando la complejidad que supone la programación a bajo nivel con este tipo de dispositivos. También incluye algunas herramientas para ejecutar desde nuestra línea de comandos, todas ellas herramientas de muy bajo nivel que nos permiten escribir y leer ciertas TAG-NFC, detectarlas con nuestro dispositivo NFC.

"Libfreefare" es otro conjunto de librerías que pretende brindar una API para tecnología MiFare mucho más conveniente. Está escrita en C y se aprovecha de las capacidades que nos traía "libnfc". También lleva un tiempo en estado de madurez y nos proporciona muchas más herramientas, todas ellas de nivel bastante más alto que 'libnfc', permitiéndonos leer, escribir y formatear las TAG-NFC MiFare, haciendo una correcta distinción entre MiFare Classic, Desfire y Ultralight.

Aunque "nfc-tools" cuenta con un par de herramientas más en su repertorio: ifdnfc y nfc-eventd. No vamos a hablar de ellas, puesto que siguen en estado de beta y no se ha trabajado con ellas para este proyecto.

No obstante tenemos que hablar de otro portal "nfcpy", que posee distintos módulos de Python para dar soporte a la tecnología NFC, presentando una pequeña alternativa o complemento a las herramientas vistas hasta ahora.

El módulo "nfcpy" es el más general de los tres que nos ofrecen, y proporciona las herramientas necesarias para el intercambio de información entre nuestro dispositivo NFC y un TAG. "Ndeflib" y "ndeftool" consisten en la librería y herramientas necesarias para poder codificar,decodificar y modificar mensajes NDEF, codificación comúnmente utilizada para el intercambio de información NFC entre dispositivos smart phone.

Con todas estas herramientas tendremos suficiente para poder explorar el mundo de la tecnología NFC en sistemas Linux, cubriendo todas las necesidades que pueden presentarse al trabajar con esta tecnología, ya sea leer, escribir archivos, codificarlos en NDEF y demás posibilidades. Pero lo más importante, nos permitirá estudiar el soporte de NFC en las conexiones Wifi-Direct, explotando al máximo las propiedades de NFC. Y aunque su uso quizás sea muy limitado en las pruebas, estudiaremos su funcionamiento en profundidad para alcanzar un correcto entendimiento de la tecnología, para facilitar que en un futuro exista un mayor abanico de software libre integrado para NFC.

3 Wifi-Direct

Inicialmente conocido como Wi-Fi P2P, es un estándar de Wi-Fi con el fin de conectar dispositivos entre sí, sin la necesidad de un punto de acceso. Es importante diferenciar que esta conexión es directa y sin intermediarios de ningún tipo, como ocurría en el caso de las redes Ad-Hoc que aprovechan los nodos Wi-Fi cercanos para transmitir. Wifi-Direct establece la conexión entre los dos dispositivos exclusivamente, al igual que ocurre en la tecnología Bluetooth, pero con las capacidades que nos proporciona Wi-Fi. Otra de las características que hacen de Wi-Fi Direct una tecnología distinguida, es su funcionamiento en las bandas 802.11 a,g y n, especificadas en los estándares de Wi-Fi. Esto permite a los dispositivos poder emplear ambas conexiones al mismo tiempo, además de permitir a los usuarios elegir en qué banda trabajar. Generalmente los productos con Wifi-Direct funcionan en la frecuencia de 2.4 GHz pudiendo conectarse a las bandas 802.11g/n; algunos funcionan en 5 GHz permitiendo usar 802.11a/n. Pero a día de hoy se pueden encontrar un gran número de dispositivos que emplean ambas bandas de frecuencia.

A continuación vamos a hablar de la tecnología Wifi-Direct desde un punto de vista mucho más técnico, entrando en detalle en la arquitectura del sistema: cómo funciona y cómo está diseñado. Hablaremos de los modos de funcionamiento de este sistema, entrando en detalle únicamente en los modos de operación que nos afectan a nosotros, o que veremos a lo largo del proyecto. Explicaremos más acerca de *wpa_supplicant* y Wifi-P2P, mostrando las capacidades de WPS_NFC. Por último comentaremos brevemente los diferentes servicios que ofrece Wifi-Direct y algunas de las aplicaciones basadas en esta tecnología, que han ido apareciendo estos últimos años.

3.1 Arquitectura

Las comunicaciones WiFi-Direct se realizan a través del concepto de grupo, llamado "Grupo P2P". Estos son establecidos a través de unos pasos de negociación, en los que uno de los dispositivos hace las veces de AP y los demás dispositivos actuará como clientes. Este proceso es completamente dinámico y es durante la negociación, cuando cada dispositivo acoge su rol: el AP es llamado GO P2P (Group Owner) y el cliente, cliente P2P. Una vez negociados estos roles, queda establecido el grupo, al que podrán unirse otros dispositivos como clientes P2P.

Podemos simplificar el proceso de creación de grupos P2P con el siguiente esquema, que nos muestra los pasos a seguir a la hora de establecer una conexión Wifi-Direct. Vamos a analizar cada uno de los pasos y a explicar sus características.

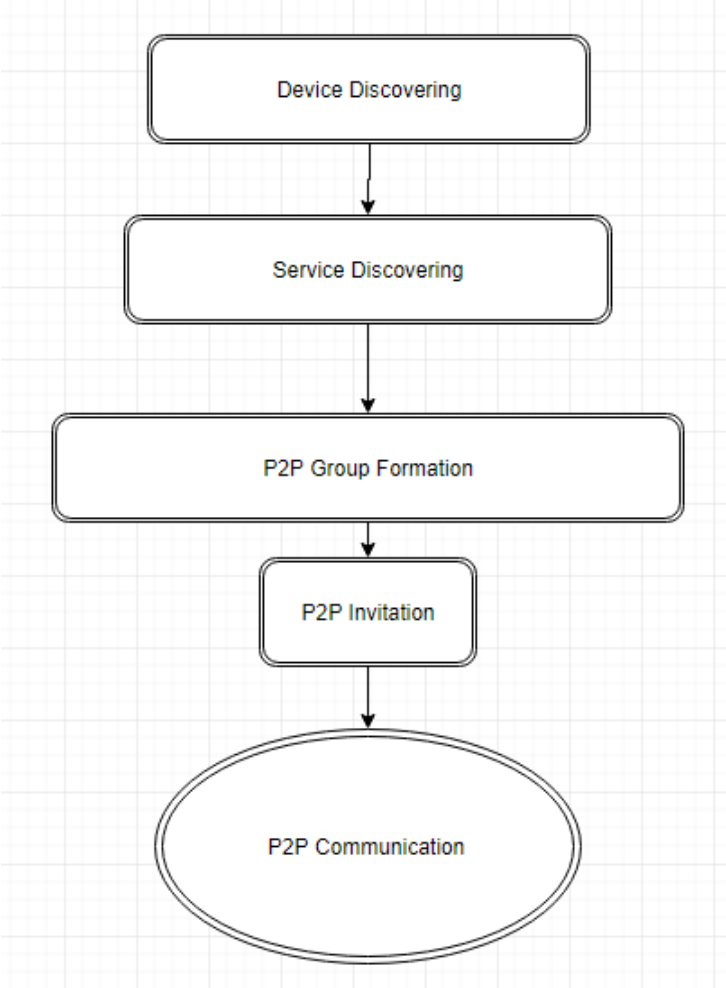

Figura 10: P2P Connection Process

3.1.1 Detección de dispositivos

Ambos dispositivos P2P realizan un primer paso de descubrimiento en el que buscan otros equipos Wifi-Direct, y al mismo tiempo se anuncian a sí mismos. Para ello escuchan y sondean alternativamente los canales 1,6 y 11 de la banda de 2,4 GHz, conocidos como "Canales Sociales". En las respuestas de ambos dispositivos se incluye información de los mismos para indicar las características que posee y las que está dispuesto a emplear en un Grupo P2P: nombre del dispositivo, tipo de dispositivo, métodos de configuración, descubrimiento de servicios, persistencia de grupo y más opciones. Las veremos de nuevo cuando hablemos de *wpa_supplicant* y los ficheros de configuración.

3.1.2 Descubrimiento de servicios

Una vez se han detectado los dispositivos, existe un segundo proceso de descubrimiento de mayor nivel que resulta opcional, la búsqueda de servicios. En este

paso se produce un intercambio de tramas que soporta diferentes protocolos para la descripción de estos servicios, véase "Bonjour" (DNS-SD), "UPnP"[7] y "WS-Discovery" [9].

Esta característica demuestran las capacidades de Wifi-Direct y las capacidades que presenta, puesto que si nos damos cuenta este descubrimiento de servicios se realiza antes de que siquiera exista una formación de grupo P2P. Los servicios se descubren en la misma capa de enlace, permitiendo al equipo o al usuario decidir si continuar con el establecimiento de conexión o no, en función de los servicios deseados.

La paltaforma de servicios de aplicación, "ASP"[9] (Application Service Platform), es la encargada de coordinar y controlar el descubrimiento de servicios entre dos dispositivos Wifi-Direct. Inicia todas las funciones necesarias para la gestión de anuncios, peticiones, las sesiones ASP, la topología y la seguridad en los servicios.

Para ello las entidades emisora y receptora se anuncian por separado empleando diferentes nombres de servicio. Al dispositivo que busca un servicio anunciado por otro, se le conoce como "servicio buscador"; al que anuncia el servicio, acoge el rol de "anunciante de servicios". Hay que tener en cuenta que un dispositivo puede verse en la situación de ambos roles, ya que pueden tener tanto anunciantes como solicitantes de servicios de forma simultanea, para diferentes entidades de servicio.

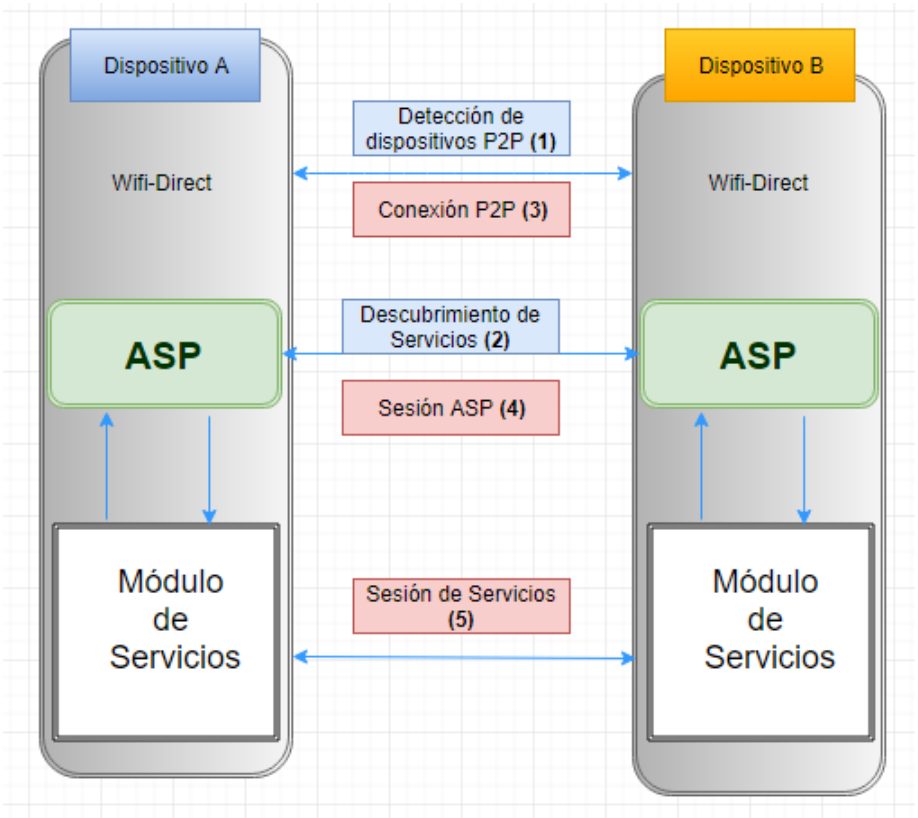

Figura 11: Establishment of an ASP session.

En la imagen se aprecia que a la hora de establecer una sesión de servicio, es la propia unidad lógica de servicios de cada dispositivo P2P, quien se comunica con su módulo ASP. Esta se encargará de mandar la información al dispositivo detectado, en caso de tratarse de una petición de búsqueda de servicio; o simplemente se encargará de recibirla en caso de pertenecer al anunciante de servicios. Tras comprobar la solicitud, se devuelve un mensaje con el nombre del servicio. Una vez establecida la conexión Wifi-Direct entre los dispositivos, se establecerá una sesión ASP que comunicará nuevamente el nombre del servicio mandado y si este es correcto dará pie a una sesión de servicios.

Es importante dejar claro que hasta no realizarse la conexión P2P, no se establecen sesiones de servicios. Y que estas pueden ser una o más sesiones de servicio con diferentes dispositivos, independientemente del rol que jueguen el dispositivos en la sesión. Este proceso puede realizarse gracias a las capacidades del módulo ASP que controlará diferentes sesiones ASP relativas al servicio buscado y emitido.

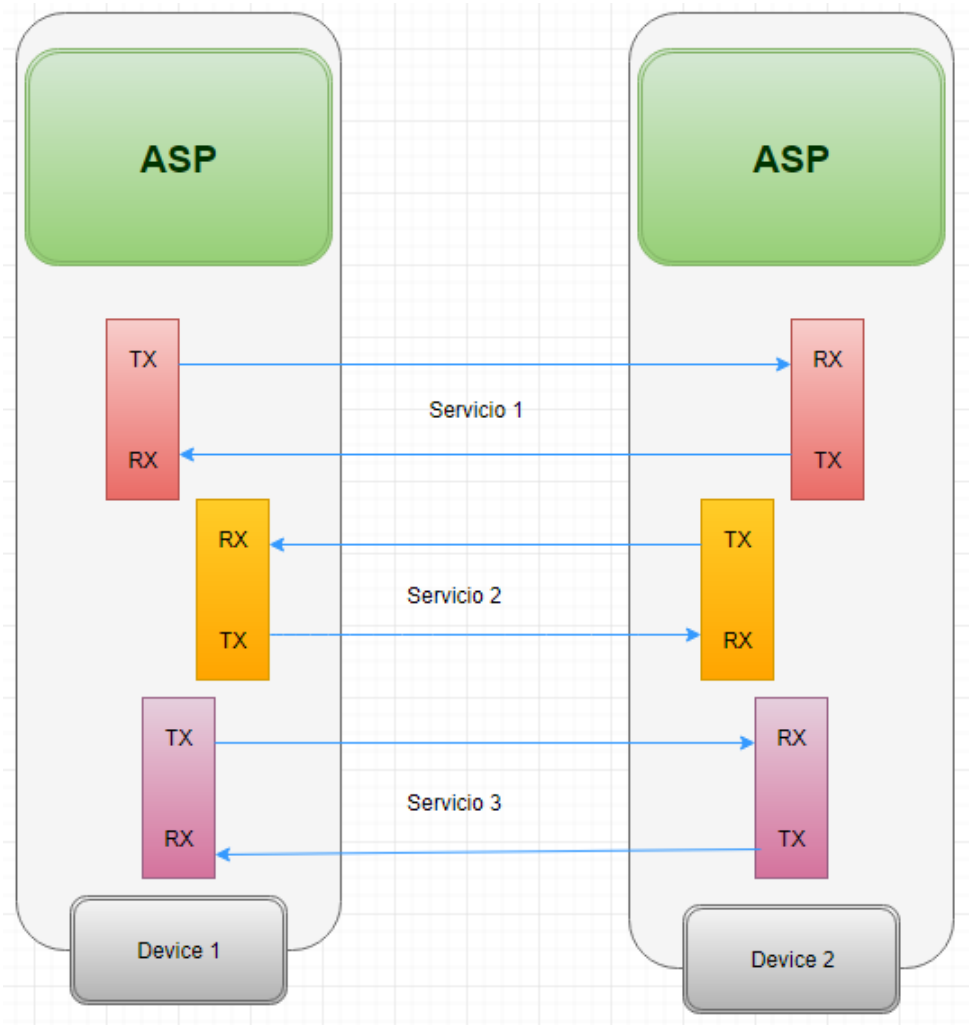

Figura 12: Services Session Transmission and Reception Example

En la figura[12] podemos entender cómo tras negociar más de una sesión ASP, al igual que veíamos en la imagen[11]. Se establecen diferentes sesiones de servicios que actúan de forma independiente y en armonía con el módulo ASP, véase figura[11].

Por supuesto el ejemplo mostrado[12] es con dos dispositivos Wifi-Direct únicamente, pero el mismo caso se daría si tratásemos con un grupo P2P en el que hubiera más equipos envueltos.

3.1.3 Formación de grupos P2P

Tras la detección de dispositivos Wifi-Direct y la búsqueda de servicios, hay que proceder a la creación de un grupo P2P, que llevaremos a cabo en dos fases indispensables.

La **primera fase** se conoce como la "fase de aprovisionamiento" o "Provisioning", en la que finalmente se establecerá la sesión del grupo P2P, para lo que los dispositivos deben realizar un intercambio de credenciales usando los métodos de configuración PIN,PBC o NFC[1]. Una vez se ha realizado este proceso correctamente, se da por finalizada la formación del grupo P2P.

Vamos a clasificar tres métodos de aprovisionamiento para la formación de grupos P2P: estándar, autónomo y persistente [2].

1. **Formación de grupo P2P estándar:**

   En este proceso ambos dispositivos comienzan un nuevo ciclo de búsqueda empleando los "canales sociales", 1, 6 y 11 de la banda de 2.4 GHz, por los que alternaran entre búsqueda y escucha. Una vez se han encontrado los dispositivos nuevamente, se procede a la siguiente fase.

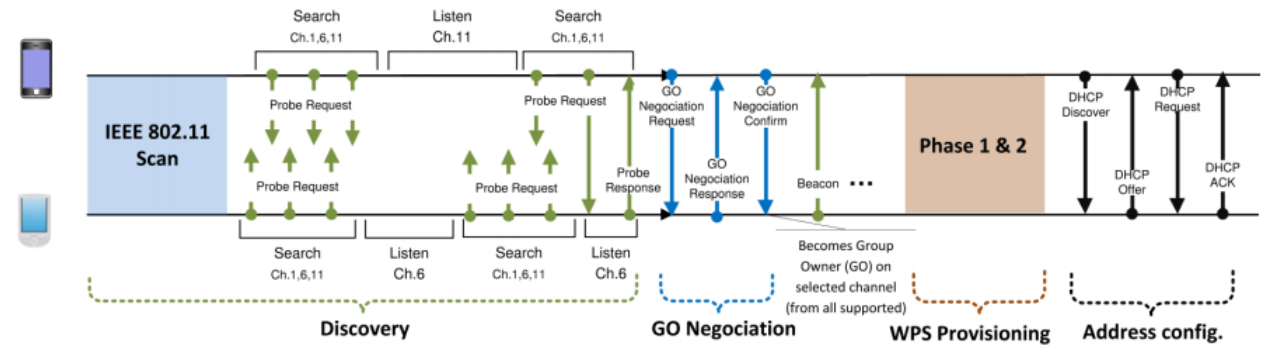

Figura 13: Standard P2P Group Formation. *Device to device communications with WiFi Direct: overview and experimentation[2], pp.3.*

2. **Formación de grupo P2P autónomo:**

   Cuando un dispositivos inicia un grupo P2P por sí mismo, es decir de forma autónoma y se declara como GO, "Group Owner" en un canal. El resto de equipos serán capaces de detectarlo en la fase de escáner, cuando busquen dispositivos P2P con los que establecer una conexión Wifi-Direct. En este caso procederían directamente sin necesidad de llegar a la fase de negociación.

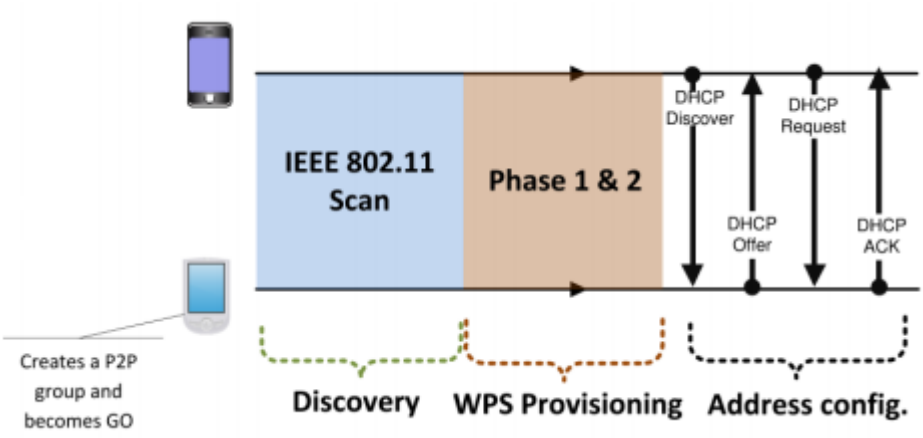

Figura 14: Autonomous P2P Group Formation. *Device to device communications with WiFi Direct: overview and experimentation[2], pp.3.*

3. **Formación de grupo P2P persistente:**

   Este último método es un caso especial en el que ambos dispositivos ya se han encontrado en un grupo P2P previamente, es decir, ya ha existido una formación de grupo estándar o autónoma anteriormente, en el que los equipos han participado.

   Gracias a la modificación de un parámetro en la formación del primer grupo, se puede definir este como "persistente" lo que permite almacenar las credenciales de la conexión P2P, incluyendo el rol que jugaban en le grupo. De esta forma cuando los dispositivos se vuelvan a encontrar durante el proceso de descubrimiento, si reconocen que ya han participado en una conexión antes, podrán restaurar el grupo P2P y asimilando cada uno su rol nuevamente. Esto reducirá significativamente todo el proceso de formación de grupo P2P, agilizando la comunicación Wifi-Direct.

   Como podemos ver en la imagen[15], tras el proceso de descubrimiento hay un pequeño paso, en el que ambos dispositivos tras haberse reconocido como miembros de algún grupo anterior, enviarán una invitación especificando la unión al grupo P2P cuyas credenciales tienen almacenadas. Lo mismo ocurre con la fase de negociación, donde el dispositivo que había sido "Group Owner" retomará su cometido. No obstante se acortará la fase de aprovisionamiento WPS de la que hablaremos más adelante.

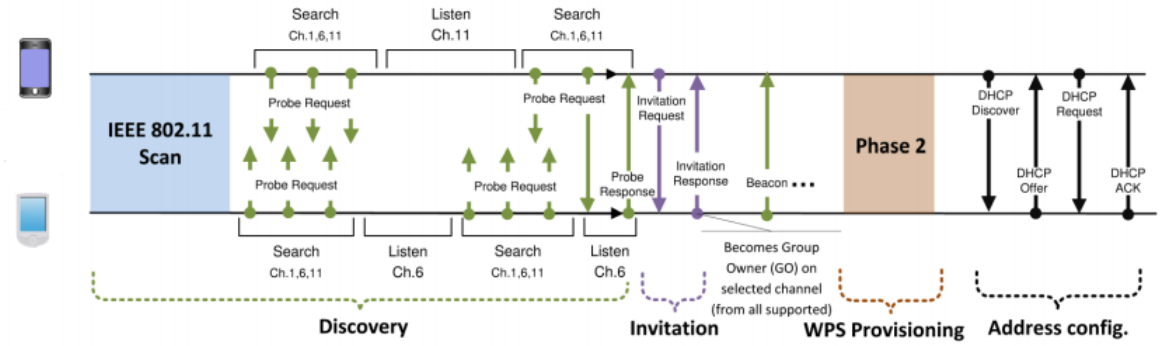

Figura 15: Persistent P2P Group Formation. *Device to device communications with WiFi Direct: overview and experimentation[2], pp.3.*

La **segunda fase** se caracteriza por la determinación del "Group Owner", quién resultará como propietario del grupo; para ello los dispositivos entablan una negociación comparando sus capacidades y seleccionando al que tenga más prestaciones, lo que atribuye el nombre de "Go Negotiation" a esta fase. El proceso de comparación se realiza con un intercambio de mensajes en la forma de "Solicitud/Respuesta/Confirmación", donde aparece el parámetro "Go Intent" que muestra en un valor numérico, la intención o interés del dispositivo por ser "Group Owner".

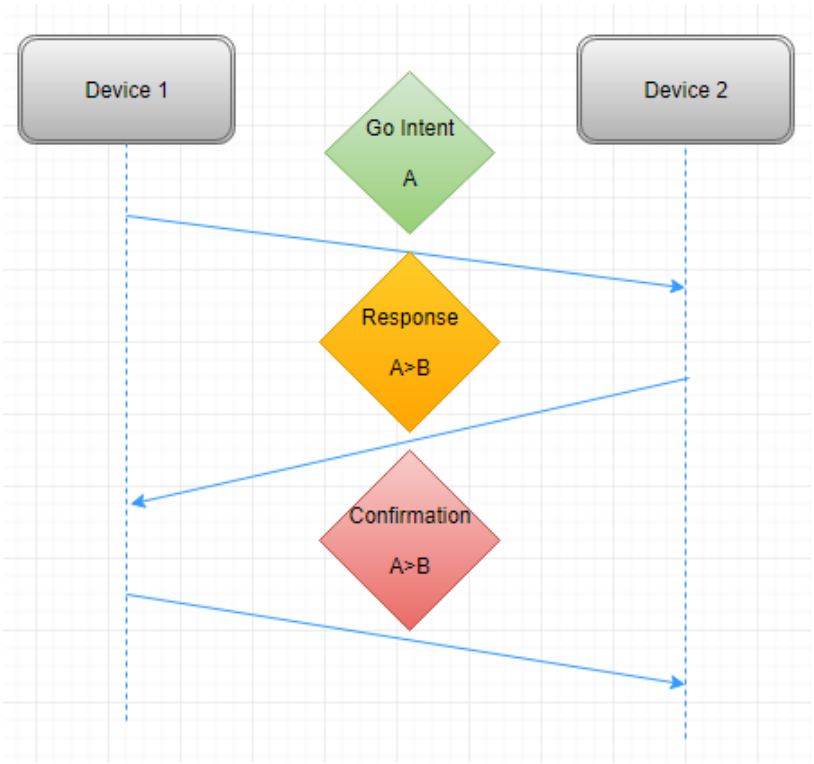

Figura 16: Negotiation Process

En la imagen[16] vemos una simplificación de cómo funciona el proceso de negociación. Cuando los dos dispositivos inician el proceso de formación de grupo,

intercambian el valor del parámetro "Go Intent" y lo comparan, el dispositivo que tenga la "intención" más alta será confirmado como el "Group Owner" de la sesión P2P. Sin embargo existe la posibilidad de que ambos dispositivos muestren la misma "intención", se elegirá uno de los dos al azar. Es importante saber que el valor de "Go Intent" puede ser configurado previamente en el dispositivo, de forma que "Go Intent" sea menor que 15. Esto lo volveremos a ver más adelante cuando hablemos del fichero de configuración en *wpa_supplicant*.

Sin embargo debemos percatarnos y entender que pueden darse situaciones donde esta negociación sea no exista o sea trivial, como son los casos de las figuras [14] y [15] respectivamente. Donde encontramos que no existe esa negociación durante el aprovisionamiento, porque uno de los dispositivos ya está definido como "GO". En el segundo caso la negociación es completamente trivial, puesto que retoman el rol que habían asumido en la primera formación del grupo P2P, donde sí habría una negociación real.

3.1.4 Invitación P2P

Este último proceso es opcional en sustitución a la formación de grupo P2P. Ya hemos visto un caso[15] en el que aparecía el proceso de invitación P2P.

La idea de la invitación P2P, es como su propio nombre indica, es invitar a otro dispositivo Wifi-Direct a que se una al grupo[1]. Por tanto podemos encontrarnos dos usos de este método:

- **Invitación desde un grupo P2P:**

  Cuando el "Group Owner" de un grupo P2P manda una invitación P2P a otro dispositivo que no pertenece al grupo, este dispositivo debe responder aceptando o rechazando la invitación. En caso de aceptarla, se incorporará al grupo como cliente P2P siguiendo un esquema de formación similar al de un grupo autónomo[14]. De igual manera un cliente P2P puede mandar una invitación a otro dispositivo, a unirse al grupo P2P del que es miembro.

- **Invitación de fuera de un grupo P2P:**

  Este caso se trata del mismo que la imagen[15], en el que un dispositivo puede mandarle a otro una invitación P2P, solicitando invocar un grupo P2P persistente. Cuando es el "GO" quien manda la invitación al cliente P2P, será este quien levante el grupo P2P nuevamente. En caso de tratarse del cliente P2P quien manda la invitación, habrá de esperar a que sea el "GO" el que comience el grupo.

3.2 WPS

La seguridad en Wifi-Direct al igual que en los procesos vistos, envuelve diversas entidades con diferentes papeles y acciones a realizar, que son negociadas al principio de este proceso.

WPS no es un mecanismo de seguridad en sí mismo, sino que define varios protocolos junto con WPA2, el estándar de encriptación avanzada AES-CCMP y claves pre-compartidas,PSK, generadas aleatoriamente con el fin de la autenticación mutua[13]; todo ello para facilitar la configuración de una red inalámbrica segura, reduciendo lo más posible la intervención del usuario. En particular WPS implementa métodos para el establecimiento seguro de conexiones como la introducción de un pin generado por uno de los dispositivos, o el pulsar un botón en ambos equipos.

La arquitectura de WPS consta de tres figuras que se definen en la red e intervienen en el proceso[13]:

- ***Registrar***: el dispositivo con autoridad para proveer o revocar las credenciales a una red. Este rol puede pertenecer al AP de forma integral o tratarse de un dispositivo independiente, pudiendo haber más de uno en la red.
- ***Enrollee***: el dispositivo que solicita acceso a la red.
- ***AP***: se trata de un punto de acceso que actua de proxy entre el *Registrar* y el *Enrollee*.

En Wifi-Direct, el proceso de aprovisionamiento WPS se inicia una vez los dispositivos se han descubierto mutuamente. El "Group Owner" hará las veces de *Registrar*, mientras que el "P2P Client" será el *Enrollee*[2]. El proceso de WPS se compone de dos fases.

En la **primera parte** el *Registrar*, en nuestro caso el GO P2P, se encarga de generar y ceder las claves de seguridad para nuestro *Enrollee*, o Cliente P2P.

A continuación[17] podemos ver en detalle la forma que presenta el intercambio de mensajes durante el proceso WPS en esta primera fase donde tras solicitar el inicio del proceso, existe una asociación para poder llevar a cabo el resto de mensajes y el intercambio de las claves hasta acabar desasociandose o desconectandose del *Registrar*.

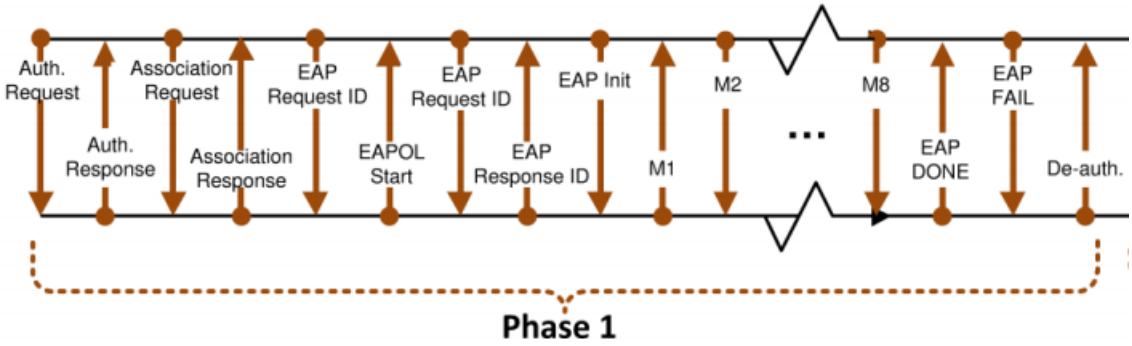

Figura 17: WPS phase 1.*Device to device communications with WiFi Direct: overview and experimentation[2], pp.3*

En la **segunda parte** el *Enrollee* o Cliente P2P, se reconectará a la red tras desasociarse del *Registrar*, accediendo así a la misma.

En la siguiente imagen[18], vemos cómo se repite el principio del proceso estableciendo una conexión y asociación con el *Registrar*, pero en este caso el intercambio que se produce es el de las propias claves que se nos han cedido.

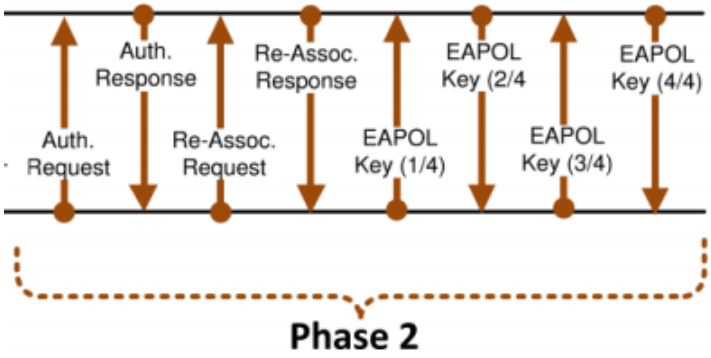

Figura 18: WPS phase 2.*Device to device communications with WiFi Direct: overview and experimentation[2], pp.3*

Esta fase[18], es la misma que veíamos directamente cuando hablábamos de la formación de un grupo P2P persistente[15], puesto que en esa situación los equipos ya han intercambiado las claves previamente, es decir que ya han realizado la primera fase anteriormente.

En Wifi-Direct podemos tratar estas dos fases WPS como: fase de autenticación, asociación y generación del PIN; fase de 4WAY-Handshake[3]. WPS contempla cuatro métodos de configuración para el intercambio de clases, de los cuales tres están implementados y se consideran en el estándar de Wifi-Direct.

- **PIN**: el usuario deberá introducir un pin conocido por ambos dispositivos. Esto requiere una interfaz con la que el usuario pueda interactuar, en general pantalla y teclado.

- **PBC**: cuando se presiona un botón , físico o virtual, en ambos dispositivos, se desencadena el proceso de intercambio de claves. Cabe decir que en el espacio de tiempo que existe una vez se ha presionado el botón en el *Registrar* de la red hasta que es pulsado en el *Enrollee*, cualquier otro dispositivo podría ganar acceso.
- **NFC**: se emplea la tecnología NFC como medio para el intercambio de credenciales. Esto requiere de una interfaz con capacidades NFC. Este método comenzó a tener soporte para Wifi-Direct en el año 2014[16].

3.3 Servicios de WiFi-Direct

Los servicios en Wifi-Direct resultan un complemento que hace que su tecnología resulte más utilizable, permitiendo llevar a cabo un serie de tareas que los desarroladores pueden explotar a su gusto. Existen cuatro servicios definidos por el estándar[9] y que se incluyen en la especificación de WFDS[10], y que permite tener varias sesiones ASP al mismo tiempo [3.1.2]. La arquitectura de WFDS se puede entender de la siguiente forma.

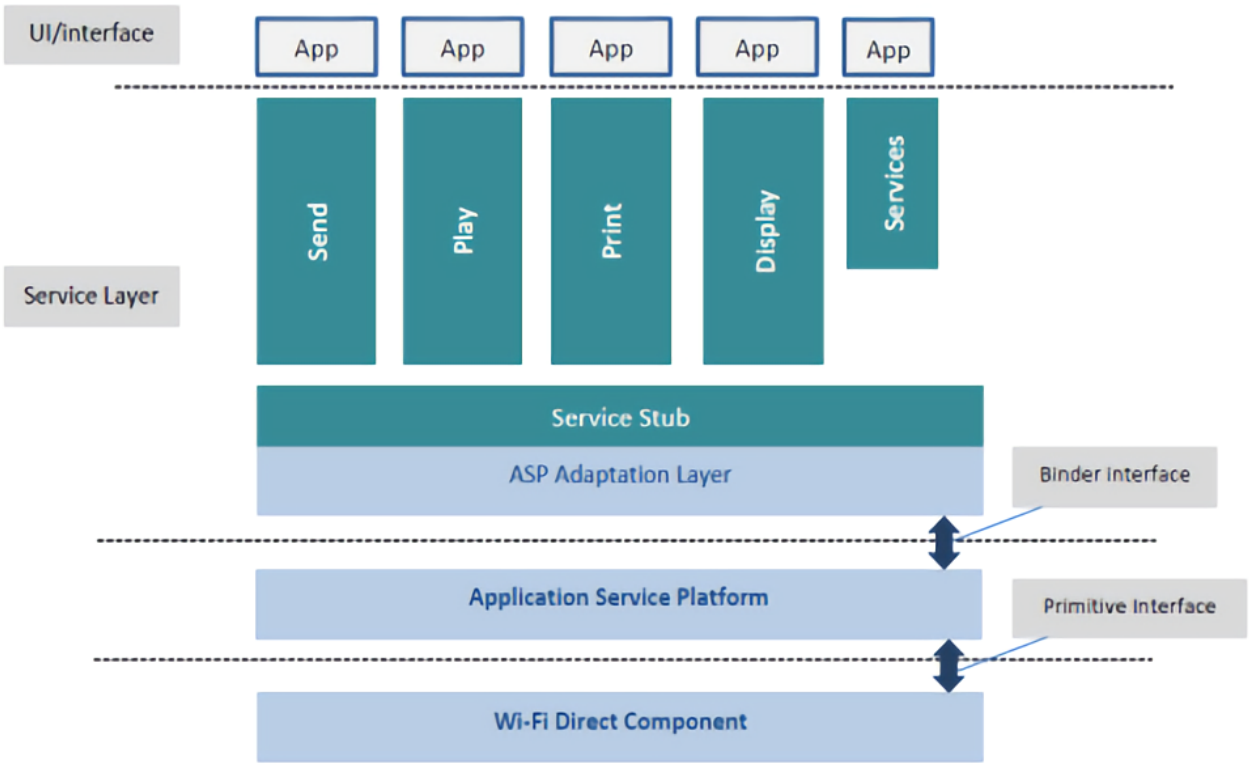

Figura 19: WFDS and Overall Architecture. *WFDS-A new Emerging technology over Wi-Fi Direct, pp.1, 2014.*

En la imagen[19] podemos apreciar los cuatro servicios que se implementan en Wifi-Direct.

- **Send Service:**

  Este servicio permite la transferencia de datos en bloque entre dos dispositivos. Para ello se emplea "HTTP" como protocolo de gestión de la transferencia de datos, resultando el emisor como cliente HTTP y el receptor de los datos, como servidor HTTP.

- **Play Services:**

  A través del establecimiento de una conexión DLNA[10], este servicio permite al usuario reproducir contenido multimedia en otros dispositivos, brindándole una interfaz de control. Como por ejemplo, vídeos en otros dispositivos gracias al renderizador "DLNA", que tiene capacidad para soportar los sistemas de codificación y compresión "H.264", "JPEG", "LPCM", "MP3" y "AAC", entre otros[10].

- **Print Services:**

  Este servicio permite la impresión a través de Wifi-Direct. Basado en el protocolo "IPP", "Internet Printing Protoco", implementa un modelo "cliente/-servidor" mediante una sesión. Donde el "cliente" es el dispositivo que desea imprimir, y el "servidor" es la impresora que recibe los datos de impresión a través de la sesión IPP[8].

- **Display Services:**

  Este servicio aprovecha las posibilidades de WFDS[10] y está contemplado como la especificación técnica de Miracast™ [11]. Permite comunicar todos aquellos dispositivos que no sean compatibles con WFDS, ni ASP[3.1.2], con todos aquellos que sí. Con el fin de lograr visualizar la pantalla de uno de los dispositivos, en el otro. Como puede ser el caso de visualizar el contenido del dispositivo smart phone en una smart TV.

Algunas aplicaciones notables que podemos encontrar basadas en estos servicios son Miracast™[12] de WiFi-Alliance y AirPlay™ de Applle. Ambas con el objetivo de poder mostrar el contenido del móvil en la pantalla de la TV. Para ello AirPlay™ utiliza tanto tecnología Wi-Fi, como WiFi-Direct y Bluetooth. Mientras que Miracast™ emplea WiFi-Direct únicamente y su objetivo va más allá, buscando el poder compartir contenido multimedia entre diferentes dispositivos, en alta calidad.

3.4 NFC en Wifi-Direct

Como hemos visto anteriormente en la sección 3.2, WPS cuenta con la tecnología NFC como uno de los métodos para realizar el intercambio de claves entre el *Registrar* y un *Enrollee*, lo que implica que ambos dispositivos posean un interfaz NFC y se comuniquen a través de ella de forma directa, o mediante el uso de un TAG-NFC. Esto supone que el intercambio de claves WPS de la fase 1[17] no requiere interacción por parte del usuario, más allá de aproximar de los dispositivos entre sí. Una vez producido el intercambio de claves, los dispositivos pasarán directamente a la fase 2[18] de la autenticación. Por supuesto si las claves son almacenadas en una TAG-NFC, también podrán compartirse con los *Enrollee*, sin necesidad de que estos se comuniquen con el *Registrar*, entrando directamente a la fase 2[18].

Wifi-Alliance incluyó el soporte para la tecnología NFC en el estándar de Wi-Fi Direct en el año 2014[17], que se contempló en la versión 1.3.32 del documento en cuestión[1]. En él se muestra el uso de la tecnología NFC, en dos casos muy similares que se consideran dentro del proceso de descubrimiento y detección de otros dispositivos P2P, en la forma de "Descubrimiento out-of-band".

Si un usuario reconoce que dos dispositivos P2P soportan NFC, se puede iniciar el descubrimiento de ambos dispositivos a través del contacto con NFC. A este proceso se le conoce como "out-of-band discovering"[1], o "descubrimiento fuera-de-banda", ya que literalmente no necesitan realizar un escáner para encontrarse, como es el caso habitual, simplemente es el usuario quien los encuentra y junta.

El "descubrimiento fuera-de-banda NFC" se utiliza para asegurar que dos dispositivos P2P se pongan de acuerdo en un canal donde realizar la negociación GO, y llevar a cabo la formación de un grupo P2P, si no existe ninguno en el que estén interesados[1]; durante este proceso también se intercambian las claves y atributos P2P necesarios para el establecimiento del nuevo grupo P2P, o permitir que otro dispositivo se una al mismo. Para este proceso, el estándar de Wi-Fi Direct contempla dos casos para el "Descubrimiento fuera-de-banda"[1]:

- **NFC Negotiated Handover:** Cuando dos dispositivos P2P con capacidades NFC entran en contacto con la intención de establecer una conexión P2P, se especifica a la hora de establecer la comunicación NFC-LLP, anunciándolo durante la fase de "handover" o "entrega" del protocolo NFC. Solicitando al otro dispositivo que responda con sus características incluyendo atributos P2P.

  Una vez se ha establecido la conexión, los dispositivos P2P intercambiarán los mensajes de "entrega". Estos mensajes pueden ser de "solicitud" o "selección", en los que se deberán incluir ciertos atributos de configuración del canal, credenciales y negociación, en función del rol que estén jugando. En caso de existir una colisión de estos mensajes, es el propio protocolo de "Handover" el que se encargará de gestionarla.

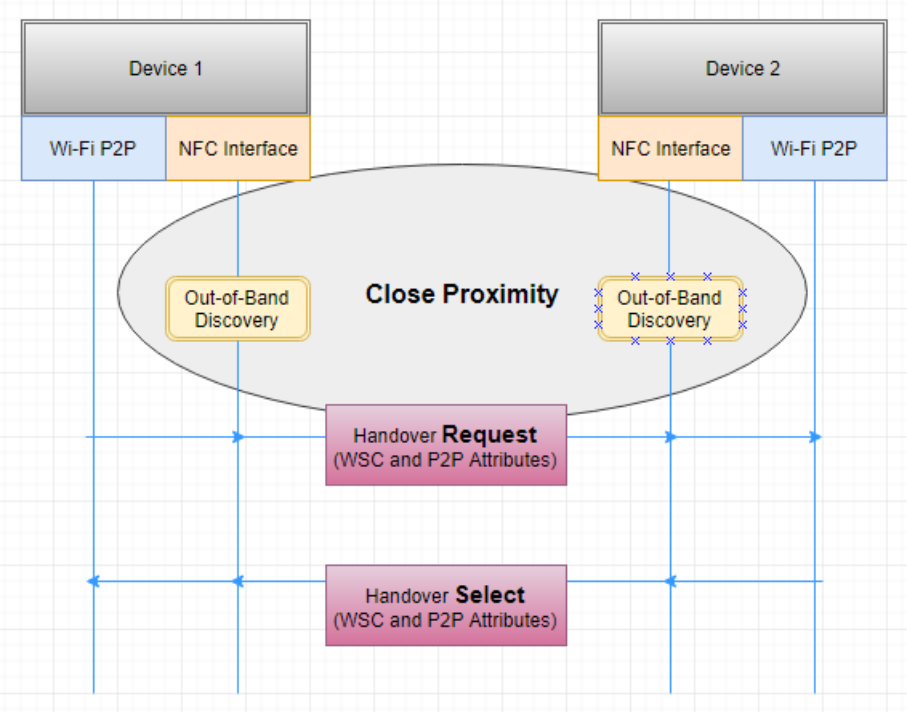

Figura 20: NFC Negotiated Handover

- **NFC Static Handover:** Este caso puede darse en algunas situaciones más concretas que envuelven algunas arquitecturas de red P2P bastante comunes. Entonces se produce una "entrega estática" cuando los dispositivos entran en contacto y es únicamente un de ellos el que realiza el "handover" mandando un mensaje de "selección" con los atributos de la conexión P2P.

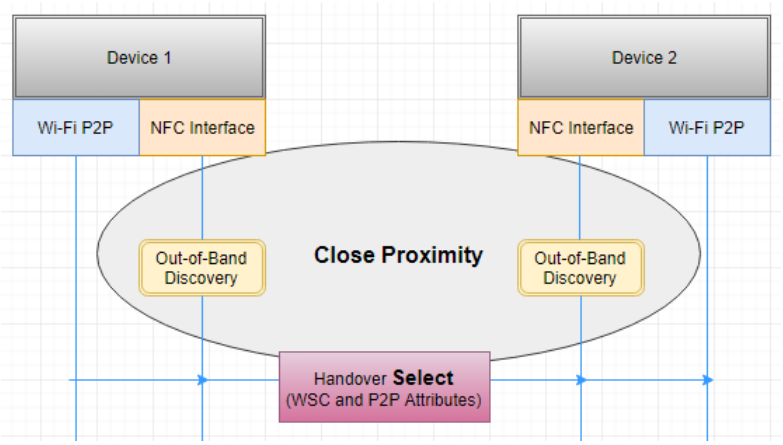

Figura 21: NFC Static Handover

Una vez se ha producido el "Descubrimiento Out-of-Band" y los dispositivos a intercambiado los parámetros P2P de alguna de las dos formas que hemos visto,

"Negotiated Handover"[20] o "Static Handover" [21], debemos proceder con la conexión P2P.

Vamos a ver las tres posibles arquitecturas de red P2P que pueden originarse a la hora de completar este "in-band procedure"[1], "procedimiento en-banda".

La **primera situación** se produce cuando dos dispositivos P2P no pertenecen a ningún Grupo P2P; entonces es el dispositivo que recibe el "handover" con el mensaje "Select", el encargado de establecer la comunicación usando los atributos P2P especificados para la correcta conexión. En la imagen[22] vemos cómo

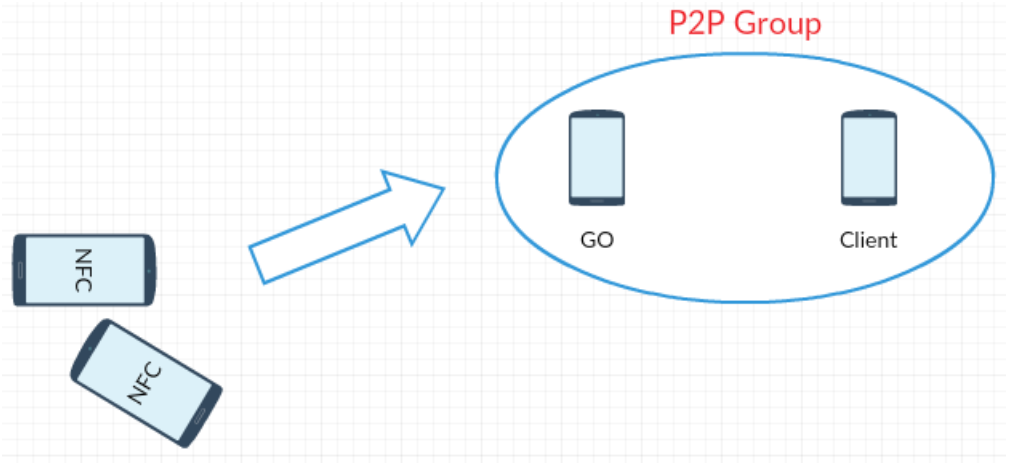

Figura 22: P2P Group Establishment through NFC

dos dispositivos P2P con interfaz NFC, entran en contacto para llevar a cabo el "descubrimiento fuera-de-banda" y al no pertenecer ninguno a un grupo P2P, es establecido por los mismos formando una red P2P simple. En este caso los dispositivos realizarían un "Negotiated Handover" como veíamos en la figura[20]. Y podría considerarse el entorno más sencillo a la hora de una conexión P2P.

La **segunda situación** ocurre cuando un dispositivo P2P establecido como "Group Owner" en un grupo P2P, entra en contacto con otro dispositivo P2P mediante interfaces NFC.

Como podemos observar en la figura[23], cuando un dispositivo que resulta ser GO de un grupo P2P, entra en contacto con otro dispositivo con capacidades NFC. Se produce un "Static handover" como en la imagen[21]. Donde el dispositivo establecido como "Group Owner" únicamente manda un mensaje "Select" durante el "handover", especificando los atributos P2P del grupo en el que es GO, de forma que el dispositivo pueda unirse al grupo como "Cliente P2P".

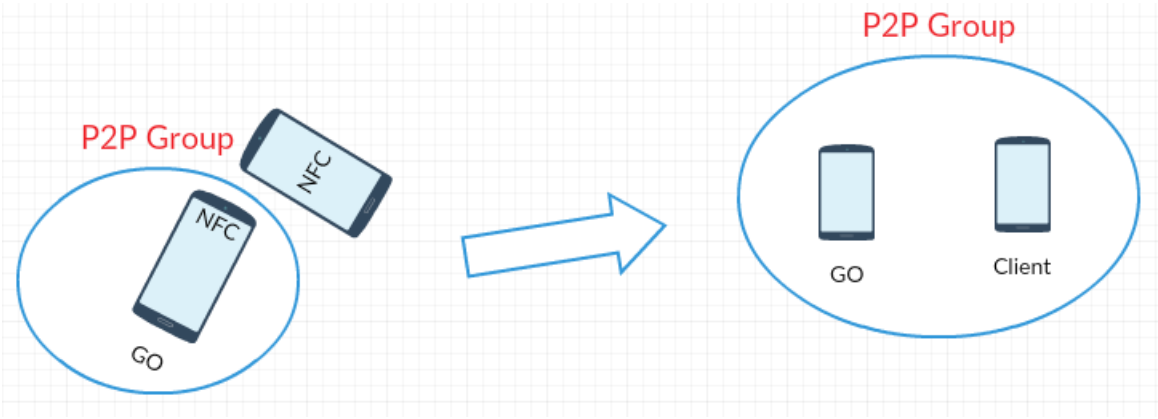

Figura 23: Joining a P2P Group with GO invitation through NFC

La **tercera situación**, muy similar a la anterior, se produce cuando un dispositivo que pertenece a un grupo P2P, entra en contacto con otro dispositivo con un interfaz NFC. En este caso los dispositivos iniciarán una "Negotiated Handover" como en la figura[20], en la que el dispositivo que no pertenece al grupo P2P, mandará la "solicitud". Esta será contestada con un "Select" que contendrá características del grupo e información para poder comunicarse con el "Group Owner", dejando a implementación la decisión de si puede o no unirse al grupo P2P[1].

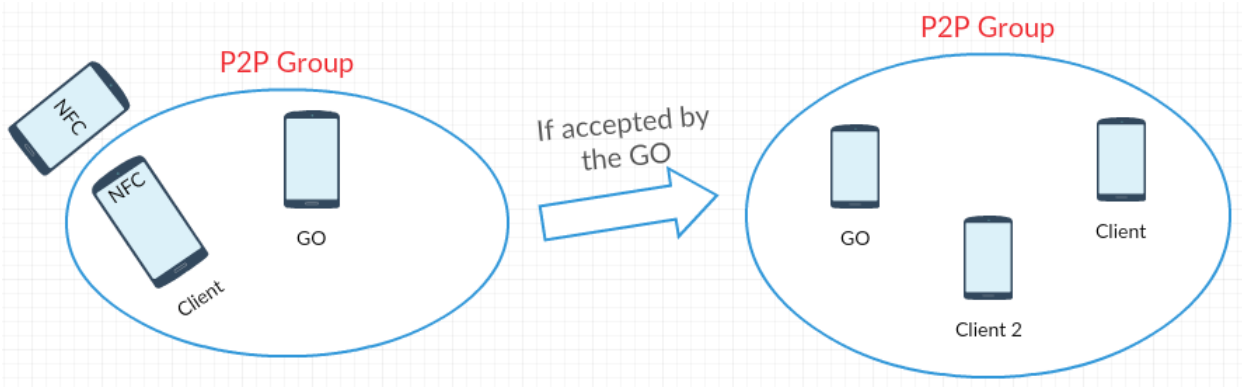

Figura 24: Requesting to Join a P2P Group with a Client invitation through NFC

La arquitectura de red P2P que presenta la imagen[24], comprende una distribución más o menos compleja, en la que podemos hacernos a la idea de las capacidades de Wi-Fi Direct. Podemos imaginarnos la red de un hogar, en la que existe un grupo P2P con un GO centralizado al que sólo tiene acceso un miembro de la familia, de forma que si alguno de los otros miembros comparten la información de la red vía un dispositivo o una TAG-NFC. No podrá unirse al grupo P2P, a menos que el GO lo autorice.

De igual forma podemos encontrar un grupo P2P donde el GO ha escrito la información del "handover" en una TAG-NFC de forma que pueda compartirse entre los usuarios para conectar sus dispositivos de forma sencilla y fiable.

3.5 Wi-Fi Direct y *wpa_supplicant*

Hasta ahora hemos hablado y visto el funcionamiento de Wi-Fi Direct a través de su arquitectura, servicios y demás componentes que aparecían en el propio estándar[1]. Todo ello aunque sin entrar en mucha profundidad, ha servido para ayudarnos a entender los entresijos de esta tecnología. Por lo que a continuación nos toca explicar y tratar de comprender la herramienta con la que vamos a establecer nuestras conexiones Wi-Fi Direct.

Wpa_supplicant es una implementación en software del componente "WPA Supplicant" que se ejecuta en el cliente de una red. Lleva a cabo la negociación de claves WPA con un autenticadorr WPA y autenticación EAP con un servidor de autenticación[18]. Está diseñado como un proceso "demonio" que se ejecuta en segundo plano, actuando como soporte para las comunicaciones inalámbricas. *Wpa_supplicant* funciona en los kernel Linux 2.4.x o 2.6.x[18], lo que permite su ejecución en una gran variedad de dispositivos basados en esta tecnología. Adicionalmente *wpa_supplicant* implementa la norma IEEE 802.11i/WPA2, así como WPS, pudiendo configurarlo para llevar a cabo comunicaciones como "Registrar" y "Enrollee". Sin embargo nosotros vamos a explorar la implementación del estándar de Wi-Fi Direct en *wpa_supplicant*, para lo que nos aprovecharemos de las capacidades P2P y WPS que presenta.

A la hora de iniciar *wpa_supplicant* debemos seleccionar un fichero de configuración en el que incluiremos las características que queramos que tengan nuestra conexión y dispositivo. En el caso de Wi-Fi Direct, deberemos añadir los parámetros que nos interesan para nuestro equipo, como el nombre, el tipo de dispositivo o los métodos de configuración WPS que vamos a emplear para establecer la conexión. Uno de los parámetros indispensables a la hora de definir la configuración de un dispositivo P2P, será su "intención" con la que se negociará tras el aprovisionamiento de la conexión.

Wpa_cli es una herramienta de *wpa_supplicant* que nos permite comunicarnos con el proceso "demonio" de este, a través de comandos de texto que nos permiten saber su estado, configuración, eventos y solicitudes. Dentro de la lista de comandos que utiliza, *wpa_cli* cuenta de forma opcional con comandos para establecer conexiones Wi-Fi Direct, Wi-Fi P2P en *wpa_supplicant*. Para habilitar este componente, ha de especificarse en la propia configuración del programa a la hora de su instalación.

Dentro de las posibilidades de *wpa_cli*, también encontramos comandos para llevar a cabo opciones de WPS, en los que podemos ver claramente los métodos de PIN, PBC y NFC que emplearemos a la hora de establecer nuestras conexiones P2P desde nuestro terminal Linux.

Algunos de los comandos más notables desde el punto de vista de Wi-Fi Direct son[19]:

- **p2p_find**, que nos permitirá iniciar la fase de descubrimiento y detección de otros dispositivos P2P y alternativamente darse a conocer.

- **p2p_connect**, mediante el cual podremos solicitar una conexión con otro dispositivo P2P que hayamos detectado. Para ello necesitaremos especificar la dirección del dispositivo al que queremos conectarnos, y la configuración que queremos emplear para el intercambio de claves de WPS (PBC o PIN).
- **wps_pbc**, nos permitirá "presionar" el botón a la hora de producirse un intercambio de claves WPS mediante este método.
- **p2p_group_add**, con el que crearemos un grupo P2P de forma manual, declarando nuestro equipo como "Group Owner". También puede utilizarse para invocar un grupo P2P persistente.
- **wps_pin**, nos permitirá introducir el pin establecido por un dispositivo, para realizar la conexión con este método WPS.

Por supuesto existen muchos más comandos importantes, aquí hemos dejado reflejados únicamente los que son esenciales para poder establecer una conexión Wi-Fi Direct a través de *wpa_cli* junto con *wpa_supplicant*. No obstante hay que tener en cuenta, que esta misma herramienta se emplea para establecer las conexiones Wi-Fi, llevar un manejo y un control de las redes, o simplemente establecer una configuración en la que poder limitar sus capacidades, donde nosotros solo vamos a aprovechar la implementación de Wi-Fi P2P y su capacidades WPS.

Para terminar, es importante mencionar que dentro del conjunto de comandos WPS, aparece una lista de comandos "wps_nfc" que implementan todas las herramientas necesarias para poder realizar un intercambio de credenciales WPS, a través de la tecnología NFC[21]. En este apartado encontraremos comandos como:

- **wps_nfc**, para activar las funcionalidades de este método en *wpa_supplicant*.
- **wps_nfc_token**, que nos proveerá del token de seguridad generado por WPS y que deberá ser intercambiado por otro dispositivo.
- **wps_nfc_tag_read**, con lo que podremos leer los datos recibidos por otro dispositivo a través de NFC.

4 NFC

RFID permite la comunicación entre un dispositivo lector activo y una etiqueta electrónica pasiva sin alimentación, mediante el uso de ondas de radio. Esto se utiliza tanto para la identificación como para los sistemas de autenticación y seguimiento. En el año 2002, las compañías Sony y Philips establecen las especificaciones de una tecnología basada en RFID. Más tarde en el año 2003 este estándar es aprobado por la ISO/IEC con le nombre de NFC. Una tecnología que permite la comunicación entre dos dispositivos electrónicos cuando se encuentran en proximidad física. pero es en el año 2004, gracias a la formación NFC Forum por la mano de Nokia, Sony y Phillips, cuando la tecnología NFC empieza a ser algo prometedor. El NFC Forum es una asociación sin ánimo de lucro, cuyo objetivo es mejorar la tecnología NFC e instaurarla como una tecnología común a nivel mundial.

En el año 2006 NFC Forum presenta su arquitectura definitiva para la tecnología NFC, junto con las primeras especificaciones de las TAG-NFC, complemente indispensable de esta tecnología. Desde entonces NFC ha ido creciendo añadiendo capaciedades a su repertorio, como el protocolo Peer-to-Peer que permite transferir datos a otros dispositivos NFC, o iniciar una conexión Bluetooth.

A partir del año 2010 se empezaron a comercializar móviles con esta tecnología, abriendo un abanico de posibilidades en el mercado, que llevan al nacimiento de empresas dedicadas a esta tecnología específicamente. Se logra empleear NFC como medio para compartir información multimedia, y surge el servicio PayPass en las tarjetas MasterCard.

A día de hoy existen sistemas de seguridad basados en NFC, los pagos con móvil simulando nuestra tarjeta, y tarjetas SIM NFC. Sin embargo, y aunque es evidente que la tecnología NFC tiene cierto bagaje, esto es apenas el principio de lo que augura ser una era de múltiples conexiones y tecnologías de comunicación, para los que NFC será un pilar fundamental, bien como medio o bien como un soporte auxiliar.

Ahora vamos a hablar sobre la arquitectura de NFC, explicaremos el funcionamiento en la transmisión de datos junto con sus protocolos, veremos la tecnología que suponen las TAG-NFC, y por último hablaremos de "nfc-tools" y su funcionamiento en sistemas Linux.

4.1 Fundamentos de NFC

Al igual que su predecesor RFID, la tecnología NFC funciona mediante acoplamiento inductivo. Dos pequeñas antenas circulares generan lo que se llama "campo-cercano" en la banda de 13.56 MHz, esto permite que el campo magnético de la señal se acople inductivamente con otro dispositivo cercano que emplee NFC, pudiendo modular esta señal para establecer una comunicación[22]. Este "campo-cercano" tiene un alcance de no más de 10 cm, espacio en el que el acoplamiento inductivo también produce una transferencia de energía suficiente como para poder activar un TAG electrónico, al igual que ocurría en RFID. Por tanto en las comunicaciones NFC siempre habrá un dispositivo "Iniciador" y un dispositivo "Objetivo" [22]. En la imagen[25] vemos cómo se produce la comunicación entre

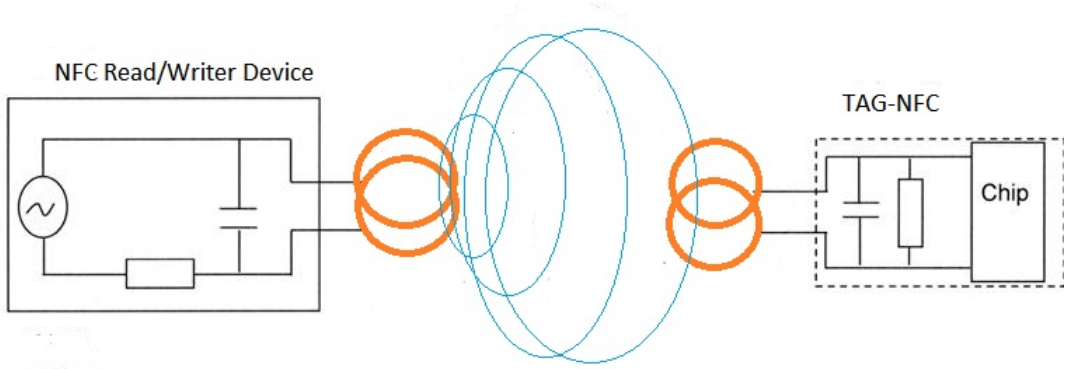

Figura 25: NFC coupling induction.

un dispositivo NFC, por ejemplo un smart phone, y un TAG-NFC. Las antenas del dispositivo smart phone generan un campo magnético que al inducir una corriente eléctrica en el TAG-NFC, nos permitirán obtener la información que este contenga. Esta tarea se realizará en velocidades de transmisión entre los 106 kbps y los 424 kbps, que se recogen en las especificaciones técnicas de NFC[24].

Las TAG-NFC resultan un complemento muy versátil para los dispositivos NFC. Se trata de un dispositivo electrónico pasivo, es decir, sin fuente de energía propia, que cuenta con un chip con capacidad suficiente para almacenar un mínimo de información. Sin embargo al tratarse de un dispositivo pasivo, siempre resultará en el "Objetivo" de las comunicaciones NFC, dependiendo del dispositivo "Iniciador" para que pueda producirse el susodicho intercambio de datos. Al mismo tiempo esto abre un abanico a la hora de la fabricación de TAG-NFC, ya que podemos encontrarlas en pegatinas, juguetes, llaveros e incluso carteles, pero sin duda la forma más extendida son las tarjetas.

Existen cuatro tipos de TAG-NFC básicos conocidos por su numeración del 1 al 4, presentando cada uno de ellos formatos y capacidades diferentes. Los TAG-NFC se conforman basados en la norma ISO 14443 A y B, estándar también de las "Contact-less Smartcards", "Tarjetas inteligentes sin contacto"[24]; además de la norma ISO 18092, implementada con las Sony FeliCa, muy extendidas en Japón y

el resto de Asia. Los TAG-NFC de tipo 1 y 2 tienen capacidad de 96 Bytes y 48 Bytes respectivamente, pudiendo funcionar tanto para lecutra o escritura, y alcanzando una tasa de transmisión de 106 kbps, por lo que las TAG-NFC de tipo 1 no eran rentables; los TAG-NFC de tipo 3, basadas en Sony FeliCA poseen una capacidad de 2 KByetes de memoria con una velocidad de 212 kbps, lo que encarece su fabricación en comparación a las otras; las TAG-NFC de tipo 4 tienen capacidad de memoria que puede llegar a 32 KBytes, con unas tasas de transmisión que alcanzan los 424 kbps. Las TAG de tipo 3 y 4 deben ser escritas en fábrica con un equipo especial, pudiendo utilizarse como "solo lectura", aunque esto supone un problema ya que a cualquier dispositivo perteneciente al NFC Forum se le requiere la capacidad de leer y escribir cualquiera de los 4 tipos[23]. Por último mencionar la existencia de un TAG-NFC tipo 5 que permite la lectura de mensajes de tipo NDEF, basada en la tecnología NFC-V[24] y permite el uso de comunicaciones activas, como abrir una página web o realizar una llamada cuando el usuario está cerca de esta.

4.2 Arquitectura de los dispositivos NFC

Una vez hemos comprendido cómo funciona NFC de forma física, junto con la existencia de las TAG-NFC y algunas de sus capacidades generales, debemos adentrarnos en los dispositivos NFC y estudiar cómo están diseñados, qué protocolos implementan a la hora de la transmisión de datos y cuáles son sus capacidades.

Con el fin de realizar una pequeña clasificación y explicar los protocolos envueltos en las comunicaciones NFC, vamos a empezar por el final, hablando de los servicios que nos proporciona NFC[24].

- **Peer-to-Peer:**
  Este modo permite el intercambio de información entre dos dispositivos. Se puede utilizar para compartir archivos, información de contacto e incluso conexiones Bluetooth.

- **Read/Write:** Con este modo un usuario puede leer y escribir información en TAGs-NFC. Un dispositivo utilizando este modo, también es capaz de leer TAGs-NFC formateadas de acuerdo a las especificaciones del NFC Forum. Esto abre la posibilidad de poder leer información de carteles, escaparates o imágenes que contengan información NFC.

- **Emulación de Tarjeta:** Como su propio nombre indica, este modo permite a un dispositivo NFC emular el funcionamiento de una "Smartcard" o "Tarjeta Inteligente", pudiendo ejecutar las mismas funciones que una TAG-NFC común. Esto permite realizar funciones como pagos a través de nuestro dispositivo NFC o simplemente simular una entrada con nuestro móvil, garantizándonos el acceso.

Todos estos servicios son el resultado de la arquitectura de protocolos que conforma un dispositivo NFC

Como podemos ver en la imagen[26], cada uno de los servicios depende de un protocolo o procesos específicos. "Peer-to-Peer" dependerá del protocolo

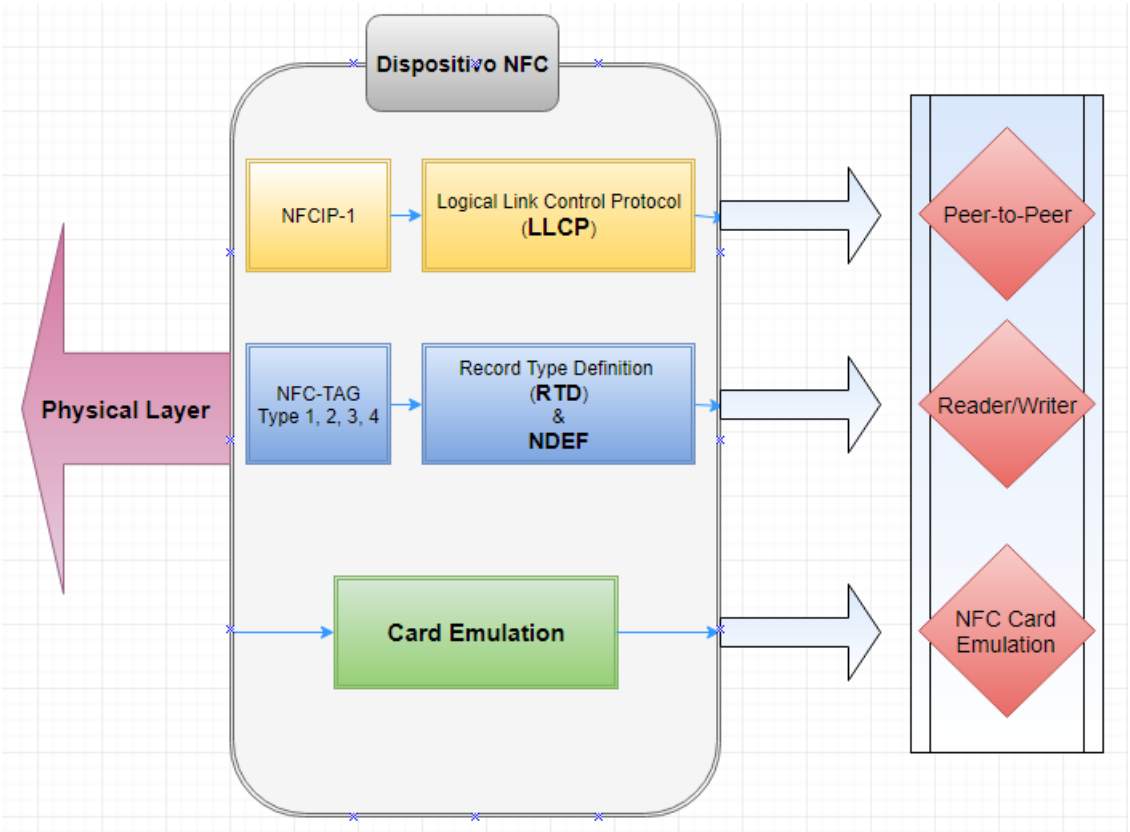

Figura 26: NFC Device Architecture

"LLCP" que nos permitirá conectar dos dispositovos NFC; los servicios de "Lectura/Escritura" se apoyarán en los métodos de codificación que emplea NFC siendo el formato NDEF el más importante, ya que nos permitirá entender la información que compartamos entre los dispositivos NFC y TAG-NFC; por último la "Emulación de Tarjetas Inteligentes" se llevará a cabo por el módulo del mismo nombre, permitiendo a nuestro dispositivo NFC actuar como una tarjeta inteligente. A continuación vamos a explicar más detalladamente el funcionamiento de LLCP y NDEF.

4.3 LLCP

El protocolo "LLCP", siglas de "Logical Link Control Protocol" o "Protocolo de Control de Enlace Lógico", brinda los medios y procesos necesarios para la transferencia de información entre dos dispositivos NFC. LLCP implementa las funciones que se realizan en la mitad superior de la capa de enlace del modelo OSI, complementándose con el resto de esta capa y la capa física, a través de un mapeado que especifica los requerimientos LLCP a un protocolo MAC definido externamente[25].

Las principales características que nos va a aportar LLCP son el control de la activación y desactivación de la comunicación de los dispositivos, encargándose establecer el enlace LLCP con los demás dispositivos NFC compatibles, supervisar

la conexión y finalizarla cuando sea solicitado. LLCP también tiene la capacidad mantener varias instancias de protocolos de mayor nivel al mismo tiempo.

Por tanto y siguiendo las especificaciones técnicas[25] podemos encontrar tres componentes dentro del módulo LLC: una parte de enlace encargada de comunicarse con el mapeado MAC; una parte de transporte que dividiremos en dos, orientada a la conexión, o no.

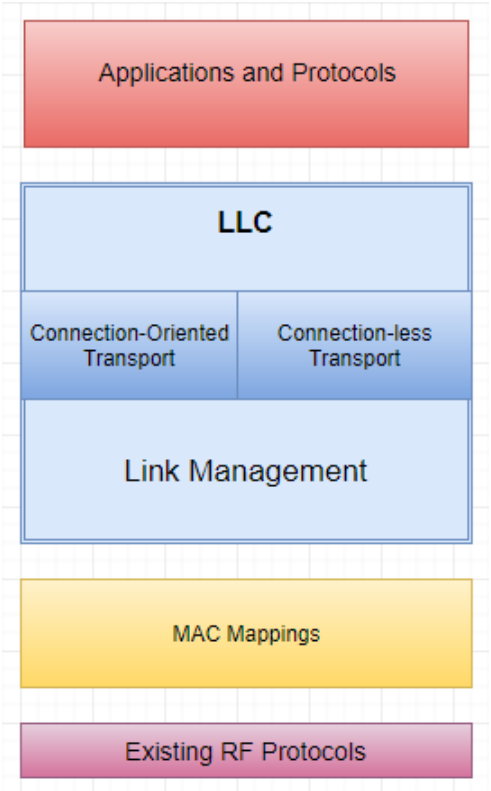

Figura 27: LLC Logical Architecture

El módulo LLC utiliza para su comunicación una estructura de paquetes denominada "LLC PDU", que podemos encontrar en las especificaciones del protocolo[25] y con ellos somos capaces de representar los procesos llevados a cabo en la capa de enlace. Podemos ver su formato en la siguiente imagen.

[h]

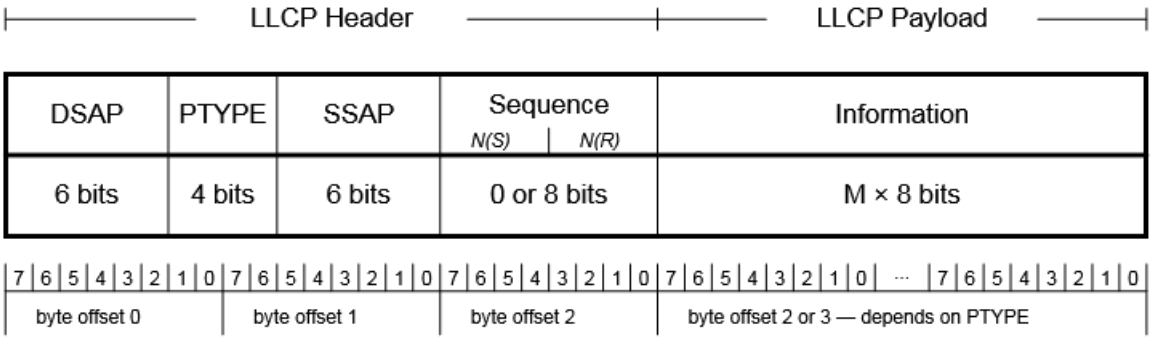

Figura 28: LLC PDU packet format. *Logical Link Control Protocol Technical Specification*[25]

- **DSAP:** "Destination Service Access Point Address Field", aquí se incluye la dirección de destino del punto de acceso al servicio.
- **PTYPE:** "Payload Data Unit (PDU) Type Field", en este campo se indica el tipo de carga que contendrá el paquete.
- **SSAP:** "Source Service Access Point Address Field", aquí se incluye la dirección de origen del punto de acceso al servicio.
- **Sequence:** "Sequence field", se incluyen 8 bits para formatos que contienen números de secuencia, y 0 bits para los que no.
- **Information:** "Information field", contiene información necesaria en cada tipo de paquete. M es un número entero que va de 0 a MIU, "Maximum information unit", "Unidad de información máxima" que por defecto es 128.

Control de enlace:

Conocido en el estándar como "Clase de servicio de enlace"[25], define dos servicios para el transporte de información en base a la orientación de la conexión. Los dispositivos que ofrecen un servicio conexión no-orientada se les clasifica como "servicio de enlace 1"; mientras que todos aquellos que ofrecen servicios de conexión orientada son clasificados como "servicio de enlace 2". Existe la posibilidad de que algunos dispositivos ofrezcan ambas posibilidades, en cuyo caso son clasificados como "servicio de enlace 3".

Cuando dos dispositivos NFC se encuentren y deseen iniciar una comunicación Peer-to-Peer, el dispositivo "Iniciador" deberá exponer el servicio de conexión que pretende utilizar. Este parámetro va escrito en el campo de información de un paquete PDU donde el "Objetivo" leerá el servicio de conexión ofrecido, siendo estos 1, 2 y 3 respectivamente. En caso de haber 0, se podrá interpretar de forma arbitraria según esté implementado en el dispositivo, o como un "servicio de enlace 3". Una vez determinado el servicio de enlace, se procederá a levantar dicho tipo

de conexión mediante el "proceso de activación de enlace", y si esta es aceptada comenzar la transmisión de datos.

Transporte de conexión no-orientada:

También llamado "transporte sin conexiónn" o "connectionless transport" en las especificaciones[25], consiste en un servicio de transmisión "unacknowledged" con la mínima complejidad posible para el protocolo PDU. Cuando hablamos de "unackowledged", literalmente "ignorado", nos referimos a que no recibe ningún tipo de confirmación o respuesta por parte del dispositivo enfrentado.

Este servicio resultará útil cuando en la comunicación con aplicaciones o capas más altas, aparezcan mensajes de recuperación y secuenciación, donde los mensajes prescinden de una confirmación. También se empleará cuando en una aplicación no sea necesario garantizar la recepción de información.

A nivel de paquete PDU, cada "servicio sin conexión" está definido por sus cabeceras de origen y destino. Cada combinación de estas será única y representará la conexión LLC. Esto representa una ventaja clara a nivel de protocolo, ya que este tipo de conexiones no necesitarán de ningún proceso de inicio o finalización, permitiendo así el intercambio de información sin ningún tipo de preámbulo en la comunicación.

Transporte de conexión orientada:

Este método de transporte proporciona servicio de secuencia en la transmisión, garantizando la llegada de los paquetes. A este tipo de conexiones se les conoce como "conexiones de enlace de datos" en la especificación de LLCP[25].

Tras el establecimiento de un enlace de transmisión entre los dispositivos que pretenden utilizar un servicio, se transmiten paquetes PDU entre módulos LLC, conteniendo estos información del servicio y un número de secuencia para poder llevar a cabo un control de la conexión. Por supuesto, todos los paquetes deben ser respondidos con un paquete PDU de "confirmación".

Todas las conexiones están identificadas de forma única a traés de sus direcciones de origen y destino, incluidas en las cabeceras. Estas conexiones sufrirán un proceso de "finalización" cuando el usuario así lo determine, llevando así a cabo el cierre de la conexión a través de un intercambio de paquetes PDU.

4.4 NDEF

El "NFC Data Exchange Format" o "Formato de Intercambio de Datos NFC", es la forma estandarizada que implementa la tecnología NFC para llevar a cabo el intercambio de información entre dispositivos o TAGs-NFC. Estructurado en formato de mensajes binarios, NDEF encapsula el contenido definido por las aplicaciones en un único mensaje, de forma que los contenidos se pueden describir

según su tipo, longitud del contenido, y un identificador de forma opcional[26].

NDEF se trata únicamente de un formato de mensajes, por lo que no vamos a hallar ningún parámetro de conexión o dirección alguna. Según el tipo de mensaje NDEF, podemos encontrarnos URIs, mensajes de tipo MIME, o de algún tipo específico NFC. Estos últimos se utilizan para encapsular información específica de aplicaciones que pertenezcan al NFC Forum. En cuanto al contenido del mensaje, solo añadir que pueden contener otros mensajes NDEF dentro o de forma encadenada, lo que hace necesario el uso del identificador de contenido.

Los mensajes NDEF están formados por lo que se conoce como "NDEF Records", pudiendo haber uno o más de tamaño variable. Cada uno de estos "NDEF Records" se define por los atributos que veíamos antes: tipo, longitud del contenido, identificador opcional. Lo que facilita la encapsulación y el desglose de datos de gran tamaño, en varios "NDEF Records".

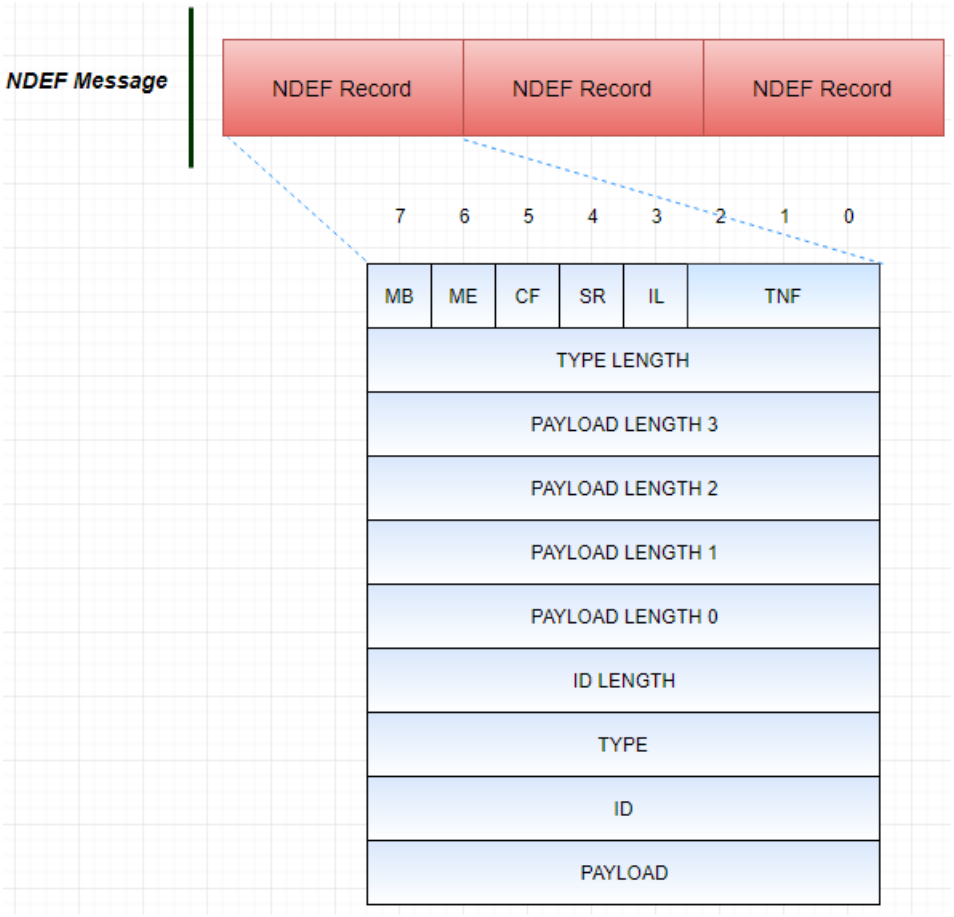

Figura 29: NDEF Message Format

A su vez, NDEF también implementa una versión más pequeña llamada "NDEF Short Record" , en donde sólo se encuentra un campo para "Payload Length", es decir un solo octeto y se especifica en la cabecera "SR" del mismo. Este formato está pensado para llevar a cabo una encapsulación más compacta o mensajes con poco contenido, de 0 a 255 octetos [26]. Los mensajes NDEF pueden contener am-

bos tipos de "NDEF Record" mezclados.

Las cabeceras MB (Message Begin) y ME (Message End), de 1-bit. indican el comienzo y el final del mensaje NDEF respectivamente. Cuando la cabecera ME está a 1, indica que ese "NDEF Record" es la terminación del mensaje NDEF. En el caso de tratarse de una parte del contenido, ME indicará únicamente el final de esa parte del contenido. Para ello debe activarse la cabecera CF (Chunk Flag), que indica que el "NDEF Record" es el principio de un contenido dividido, o alguna de las partes de en medio.

La cabecera TNF (Type Name Format), consiste en un campo de 3-bit que indican el tipo de mensaje NDEF. Los tipos de mensaje están indicados en el estándar[26] y algunos de los tipos que podemos encontrar van desde: URIs, tipo multimedia, aplicaciones conocidas del NFC Forum, aplicaciones NFC externas, tipo desconocido, o simplemente vacío, entre muchas otras.

Para finalizar hay que añadir que NDEF es una de las piedras angulares de la tecnología NFC, ya que nos aporta el lenguaje de comunicaión para nuestros dispositivos NFC. El servicio de "Lectrua/Escritura" se fundamenta en este formato para poder entender la información y transcribirla entre dispositivos. Pero también el servicio de "Peer-to-Peer" se apoya en NDEF para hacer entendible el contenido de sus intercambios, permitiendo a los dispositivos NFC compartir archivos de todo tipo.

4.5 MiFare y FeliCa

MiFare y FeliCa son las dos tecnologías presentes en el mercado de las "Smart Card". Siendo MiFare la más comercializada y común de las dos, mientras que FeliCa de Sony, ganó su mercado en Japón, otros países de Asia y Estados Unidos.

FeliCa: Diseñadas como una "Contact-less Smartcard" para la tecnología RFID, fueron propuestas para la norma ISO/IEC 14443 que definía cómo implementar las TAG-NFC, pero nunca fueron aceptadas. Sin embargo y al estar basada en tecnología RFID, es compatible con algunos estándares de modulación empleados por NFC. La tecnología FeliCa funciona con tasas de transmisión entre los 212 kbps y los 424 kbps.

Según el estándar podemos distinguir tres tipos de "Smartcard" en función del mecanismo de encriptación que sporte: tarjetas DES, tarjetas AES con claves de 128 bits y tarjetas DES/AES con claves de 128 bits[27]. Con el fin de aumentar la seguridad, la tecnología FeliCa genera las claves de forma dinámica cada vez que se produce una autenticación mutua.

La tecnología FeliCa se encuentra en varias aplicaciones como tarjetas de pago, tarjetas de acceso, pequeños llaveros, o simplemente dispositivos electrónicos como relojes inteligentes.

MiFare: Basada en tres de las cuatro partes definidas por la norma ISO/IEC 14443, MiFare presenta una tecnología con gran variedad en cuanto a las TAG-NFC. Esta tecnología soporta los sistemas de encriptación AES y DES/Triple-DES; cada tarjeta está divida en secciones o módulos, divididos a su vez en cuatro bloques. Cada sector utiliza dos claves de acceso conocidas como "A" y "B"[28], que se intercambiarán con el dispositivo lector a la hora de iniciar la comunicación.

Dentro de las tarjetas FeliCa podemos encontrar[29]:

- **Mifare Classic:** se trata de la versión más básica y proporciona un sistema de almacenamiento de información. La memoria está dividida en secciones y presenta un mecanismo de seguridad simple, ofreciendo una funcionalidad óptima a un precio muy asequible, por lo que se puede usar como sistema de pago o como entrada electrónica. Existen las Mifare Classic 1K y 4K, que permiten 1 MB y 4 MB de almacenaje respectivamente, ofreciendo características de seguridad mucho mejores.
- **Mifare Plus:** aparece como remplazo del modelo "Classic", ofreciendo una simple mejora orientada a la seguridad, donde se incluye soporte para encriptación AES de 128 bits. En 2016 fue anunciada la versión "Plus X", donde se veía una mejora notable en los protocolos de seguridad del modelo.
- **Mifare Ultralight:** con tan solo 64 Bytes de memoria y ningún tipo de encriptación, estas tarjetas ofrecen una versión de muy bajo coste que puede usarse como desechable. Se pueden utilizar como entradas desechables para eventos, como ocurrió en la Copa Mundial de Fútbol en el año 2006. "Mifare Ultralight C", fue una versión del año 2008 que integraba memoria de 192 Bytes y encriptación Triple-DES, con el fin de evitar la clonación de las mismas. Existe otra versión mejorada desde el año 2012, "Ultralight EV1" con mayor capacidad y seguridad que su progenitora, ofreciendo gran flexibilidad a los desarrolladores . Este modelo está pensado para aplicaciones de uso limitado, como los billetes de transporte público o pases de eventos.
- **Mifare SmartMX:** diseñada con fines de seguridad y uso en múltiples aplicaciones, estas tarjetas implementan las tecnologías de encriptación AES/DES, pero también PKI y ECC, ofreciendo el máximo nivel de privacidad y protección de datos. Algunas de las aplicaciones que podemos encontrar son transacciones móviles, pagos de todo tipo, sistemas de acceso e identificación en actividades del estado como la seguridad social o la

sanidad. Una mejora de esta tarjeta es la "SmartMX2", que cuenta con la funcionalidad "Mifare FleX" que permite utilizar la tarjeta como cualquiera de los otros modelos de MiFare.

- **Mifare DESfire:** similar a "SmartMX", esta tarjeta ofrece una estructura simple de directorio y archivos, viniendo ya programada. Podemos encontrar cuatro variantes según su encriptación: una con Trple-DES y tres con AES. Sin embargo todas las variantes cuentan un acelerador AES/Triple-DES con el fin de acelerar las transacciones al máximo. El modelo "DESFire EV1" ofrece soporte para AES de 128 bits, seguido del "DESFire EV2" que entre otras características incluye una arquitectura virtual de seguridad para la privacidad. y protección contra ataques de proximidad. Estas tarjetas se utilizan para sistemas de transporte público más avanzados, como identificadores en las universidades o tarjetas de acceso y micro pagos.

4.6 NFC y 'nfc-tools'

Tras ver el funcionamiento de NFC, las posibilidades que nos ofrece y las tecnologías que hay detrás de las TAG-NFC, nos queda lidiar con su uso. Como es habitual, cada fabricante ofrece sus propias soluciones de software para la utilización del producto. Esto no es distinto en NFC, donde de manera general, podremos encontrar software específico de cada fabricante para los sistemas más utilizados.

"Nfc-tools"[30] es una comunidad formada por desarrolladores, donde se facilitan las herramientas necesarias para el uso de dispositivos NFC/RFID en los sistemas Linux. Proporcionando así una serie de librerías para el futuro desarrollo de aplicaciones de forma libre. Dentro de "nfc-tools" encontramos como indispensables las siguientes herramientas:

libnfc:
Se trata de la primera librería libre, que proporciona una API de programación para el desarrollo de aplicaciones y tareas NFC de bajo nivel. Esto supone una gran ventaja ya que cubre las tareas más complicadas que nos presenta la integración de dispositivos físicos. Por tanto 'libnfc' es fundamental si pretendemos utilizar la tecnología NFC en nuestro sistema Linux.

En la imagen[30] vemos el resultado de la ejecución de una de las herramientas que nos proporciona 'libnfc'. Cuando usamos 'nfc-list' o 'lsnfc', obtenemos en respuesta la información de nuestro dispositivo NFC, junto con información de los TAG-NFC que tengamos próximos al lector. En este caso tenemos tres TAGs, dos "MIFARE UltraLight" y un tag "FeliCa Lite". Este último es detectado como dos tarjetas debido a su arquitectura.

Aunque básicas, 'libnfc' nos proporciona aplicaciones mínimamente suficientes para realizar tareas de lectura y escritura. Además de 'nfc-list' que ya hemos visto, tenemos 'nfc-mfclassic'[**?**] que nos permite leer o escribir información de las tarjetas "Mifare Classic", similar a esta encontramos 'nfc-mfultralight' con la que podremos realizar operaciones de lectura y escritura con las TAGs "Mifare UltraLight"[31].

```
debian@debian:~$ lsnfc
NFC device: SCM Micro / SCL3711-NFC&RW
UID=04929fca824981

* NXP MIFARE UltraLight
UID=043d2aca824981

* NXP MIFARE UltraLight
        ID (NFCID2): 012e3d23ba0b9f6f
   Parameter (PAD): 00f1000000014300
   System Code (SC): 88b4
   FeliCa Lite

        ID (NFCID2): 012e3d23ba0b9f6f
   Parameter (PAD): 00f1000000014300
   System Code (SC): 88b4
   FeliCa Lite

4 tag(s) on device.
```

Figura 30: 'nfc-list' command execution

libfreefare:
Aprovechando las herramientas de bajo nivel que nos presentaba 'libnfc', 'libfreefare' proporciona una API para el desarrollo de aplicaciones de mayor nivel, incluyendo el soporte para todo tipo de tarjetas MiFare. También nos proporcionará herramientas de mayor nivel que 'libnfc' para el manejo de estas tarjetas, pudiendo realizar tareas de lectura y escritura de forma más cómoda.

```
debian@debian:~$ ntag-detect
Tag with UID 043d2aca824981 is a NTAG21x
Tag with UID 04929fca824981 is a NTAG21x
Tag with UID 012e3d23ba0b9f6f is a FeliCA
```

Figura 31: 'ntag-detect' command execution

En la figura[31], vemos la ejecución del comando 'ntag-detect' muy similar al ya visto 'nfc-list' en la imagen[30]. Como podemos comparar, aunque este comnado nos presenta mucha menos información de cada una de las tarjetas, sí es capaz de detectar que sólo hay tres de ellas.

Algunas de las herramientas que podremos utilizar son 'ntag-detect' como ya hemos visto en la imagen[31], 'mifare-classic-write-ndef' para poder escribir información usando el formato NDEF, 'mifare-desfire-read-ndef'[**?**] que nos devolverá los datos de una tarjeta modelo "DESFire" que tenga datos en formato NDEF, entre otras similare que nos permitirán trabajar con los modelos "Classic", "DESFire" y "DESFire EV1" principalmente.

5 Pruebas

A continuación realizaremos dos pruebas para comprobar las posibilidades del soporte de NFC a través de *wpa_supplicant* en Linux. Crearemos dos situaciones posibles en las que poder estudiar el uso de tecnología NFC como complemento de Wi-Fi Direct, con el fin de probar el uso ambas tecnologías en futuras aplicaciones. Ambas situaciones han sido comentadas a lo largo de este trabajo: en la primera veremos un arquitectura simple donde uno de los dispositivos establece un grupo P2P independiente y el otro dispositivo se conecta mediante NFC; en segundo lugar investigaremos una formación de red más complicada en la que se inicia una conexión Wifi-Direct con un dispositivo smart phone con uno de los terminales Linux, y la información del grupo P2P se compartirá a otro terminal Linux a través de NFC.

5.1 Conexión a un grupo P2P Autónomo simple mediante NFC

Paso 1:
Como primera medida tras iniciar nuestro terminal Linux, activaremos la tarjeta de red. Para ello comprobaremos el interfaz en el que se encuentra ejecutando el comando 'iwconfig'.

```
debian@debian:~$ sudo iwconfig
lo        no wireless extensions.

wlan1     unassociated  Nickname:"<WIFI@REALTEK>"
          Mode:Managed  Frequency=2.412 GHz  Access Point: Not-Associated
          Sensitivity:0/0
          Retry:off   RTS thr:off   Fragment thr:off
          Encryption key:off
          Power Management:off
          Link Quality:0  Signal level:0  Noise level:0
          Rx invalid nwid:0  Rx invalid crypt:0  Rx invalid frag:0
          Tx excessive retries:0  Invalid misc:0   Missed beacon:0

eth0      no wireless extensions.
```

Figura 32: Ejecutamos el comando 'iwconfig'

En nuestro caso se encuentra en la interfaz "wlan1", por lo que a continuación la levantaremos usando 'ifconfig' y después repetiremos la operación para comprobar que todo está correcto.

```
debian@debian:~$ sudo ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:55:f4:f2
          inet addr:192.168.1.49  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe55:f4f2/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:329 errors:0 dropped:0 overruns:0 frame:0
          TX packets:82 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:24625 (24.0 KiB)  TX bytes:10644 (10.3 KiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:31 errors:0 dropped:0 overruns:0 frame:0
          TX packets:31 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:3323 (3.2 KiB)  TX bytes:3323 (3.2 KiB)

wlan1     Link encap:Ethernet  HWaddr c0:25:e9:27:81:c2
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
```

Figura 33: Utilizamos 'ifconfig'

Paso 2:

A continuación deberemos deshabilitar los siguientes servicios: *wpa_supplicant*, *NetworkManager*, *DNSMasq* y *HostApd*. Este paso es crucial y se dejará un pequeño manual[B] para llevarlo a cabo y arrancarlos de nuevo una vez se inicie la comunicación. Una vez ejecutado este paso nos disponemos a arrancar el *wpa_supplicant* en nuestro interfaz "wlan1" con un fichero de configuración establecido[A].

```
debian@debian:~$ sudo wpa_supplicant -Dnl80211 -iwlan1 -c p2p.conf
Successfully initialized wpa_supplicant
```

Figura 34: Iniciamos *wpa_supplicant*

También podríamos añadir la opción "-B" al arranque, para dejarlo correr en segundo plano, pero para estas pruebas no lo vamos a hacer, ya que podremos ver todos los eventos que pasan a través del *wpa_supplicant* en esta misma interfaz. No obstante, para poder continuar deberemos abrir una nueva interfaz de consola para restablecer los servicios deshabilitados al principio de este paso [B]. Una vez hecho todo lo anterior, debemos iniciar la *wpa_cli*, que automáticamente se conectará con el interfaz en el que estemos trabajando.

```
debian@debian:~$ sudo wpa_cli
wpa_cli v2.6
Copyright (c) 2004-2016, Jouni Malinen <j@w1.fi> and contributors

This software may be distributed under the terms of the BSD license.
See README for more details.


Selected interface 'wlan1'

Interactive mode

>
```

Figura 35: Iniciamos *wpa_cli*

Esta última parte la realizaremos más adelante durante la prueba, de momento usaremos la nueva consola para controlar el *wpa_supplicant* y establecer una conexión Wifi-Direct.

Paso 3:

A continuación utilizaremos los comandos de *wpa_cli*[B] para iniciar un grupo P2P en le interfaz wlan1 de forma autónoma, es decir el propio dispositivo únicamente como "Group Owner". para ello ejecutamos el comando 'p2p_group_add'[B] que al mismo tiempo que nos facilita la información del grupo P2P creado, se hará visible para el resto de dispositivos con capacidades WiFi-Direct. No obstante no es la función que nos interesa para la realización de esta prueba, simplemente queremos que exista un grupo con para la conexión.

```
> p2p_group_add
OK
<3>AP-ENABLED
<3>P2P-GROUP-STARTED wlan1 GO ssid="DIRECT-fg" freq=2412 passphrase="VQa0F1Nv" g
o_dev_addr=c0:25:e9:27:81:c2
```

Figura 36: Ejecutamos 'p2p_group_add' para la creación de un Grupo P2P

En la figura[36], vemos cómo tras ejecutar el comando 'p2p_group_add'[B], *wpa_supplicant* nos devuelve el interfaz donde se ha creado el grupo P2P y el nombre del mismo, además de otros parámetros importantes como el rol que juega nuestro dispositivo o la contraseña.

También podemos comprobar que el grupo se ha creado correctamente mirando la ventana donde hemos dejado corriendo el *wpa_supplicant*, en la que aparecerán todos los eventos que van ocurriendo.

```
Using interface wlan1 with hwaddr c0:25:e9:27:81:c2 and ssid "DIRECT-fg"
wlan1: interface state UNINITIALIZED->ENABLED
wlan1: AP-ENABLED
P2P-GROUP-STARTED wlan1 GO ssid="DIRECT-fg" freq=2412 go_dev_addr=c0:25:e9:27:81
:c2
```

Figura 37: Evento del Grupo P2P en *wpa_supplicant*

Paso 4:

En este paso emplearemos el comnado 'wps_nfc_config_token'[B] que nos devolverá una cadena hexadecimal con información precisa del grupo P2P.

```
> wps_nfc_config_token NDEF
D2178F6170706C69636174696F6E2F766E642E7766612E777363100E006C1026000101104500094
49524543542D666710030002002010 0F000200810270040626634316330643364353635303562 35
32623335363361353332376365303533613236393639663862393564383264336432386466313464
393766633132393710200006000000000000103C000101100100020001102000 6C025E92781C210
49000600372A000120
```

Figura 38: Usamos el comando 'wps_nfc_config_token

Al utilizar el comando con la opción 'NDEF', *wpa_cli* nos devolverá la cadena de valores hexadecimales con datos adicionales para poder llevar a cabo la encapsulación en formato NDEF. Por lo que podemos escribir esta información en un TAG-NFC, mediante el uso de herramientas externas para la escritura de datos en NFC.

También existe la opción 'WPS' que nos devolverá una cadena de valores hexadecimales muy similar, pero sin los valores adicionales para realizar la encapsulación NDEF. No obstante esto no significa que no podamos escribir la cadena en un TAG-NFC, podemos seguir escribiéndola pero tendría que ir dentro de algún archivo de texto plano.

Una vez tenemos nuestro TAG-NFC, lo llevaremos a un segundo terminal Linux con una configuración similar al que estamos utilizando, pero con algunas pequeñas modificaciones que podemos ver en [B]. Para que no exista confusión en las capturas, el nuevo terminal Linux utiliza el interfaz wlan0 con una tarjeta de red versión 1.x de la empleada hasta ahora, y se le referirá como "**nuevo** Linux" al pie de las mismas. Este nuevo terminal también emplea una versión modificada del fichero de configuración de *wpa_supplicant*, pero será relevante a efectos de esta prueba.

En este **nuevo** terminal Linux, deberemos repetir los pasos 1 y 2 vistos en esta prueba, para iniciar correctamente *wpa_supplicant* en el interfaz wlan0 y correr *wpa_cli* para poder ejecutar comandos.

Tras leer el TAG-NFC en nuestro **nuevo** terminal Linux mediante alguna he-

rramienta externa, recuperaremos el valor de la cadena hexadecimal que habíamos escrito. A continuación introducimos la cadena hexadecimal en *wpa_cli* utilizando el comando 'wps_nfc_tag_read'[B].

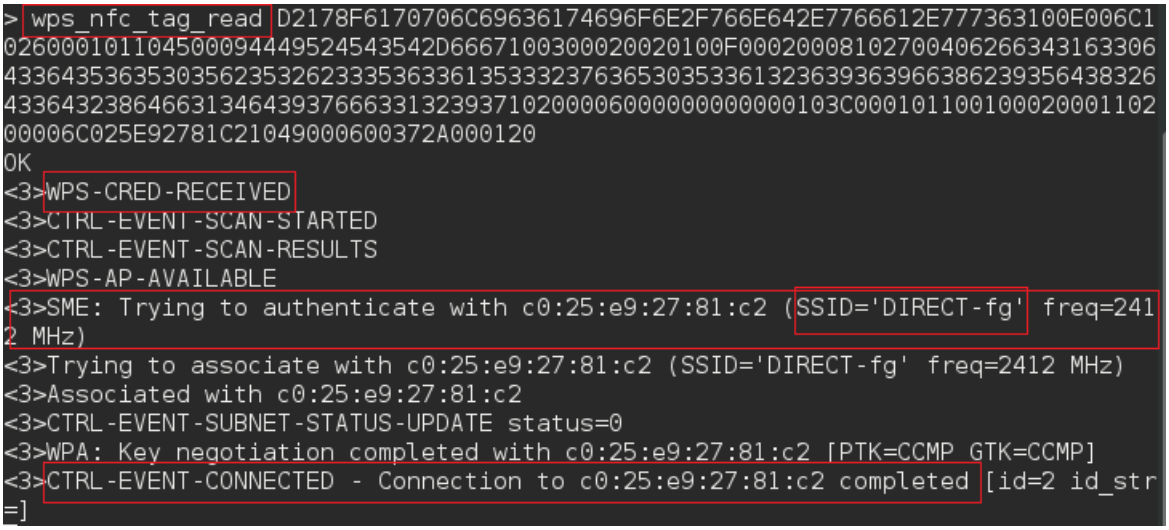

Figura 39: Introducimos la cadena hexadecimal en el **nuevo** Linux

En la imagen de la figura[39] vemos cómo tras introducir la cadena hexadecimal, *wpa_cli* nos informa de que se han recibido credenciales WPS e inicia una evento de escaneo donde encuentra un punto de acceso disponible, tras lo que intenta autenticarse y asociarse, para finalmente confirmar que se ha producido la conexión al grupo P2P.

No obstante para asegurarnos de que la conexión se ha realizado con éxito, debemos comprobar el resultado en nuestro primer terminal Linux, donde podemos mirar directamente el resultado obtenido en la terminal corriendo *wpa_supplicant*.

Figura 40: Confirmación de conexión en *wpa_supplicant*

paso 5:

Para finalizar la prueba, nos queda comprobar que ha ocurrido desde el punto de vista de *wpa_supplicant*. Ya que hemos sido capaces de compartir la información de un grupo P2P autónomo para que un segundo terminal se conecte a él directamente puesto que el intercambio de credenciales se ha producido mediante NFC.

Por tanto vamos a mirar en los ficheros de configuración de ambos dispositivos. En ellos vamos a encontrar lo siguiente.

```
network={
        ssid="DIRECT-fg"
        psk=bf41c0d3d56505b52b3563a5327ce053a26969f8b95d82d3d28df14d97fc1297
        proto=RSN
        key_mgmt=WPA-PSK
        pairwise=CCMP
        auth_alg=OPEN
}
```

Figura 41: Red almacenada en el fichero de configuración

Lo que vemos en la imagen[41] es que ambos dispositivos han almacenado la información completa del grupo P2P, como si se tratara de una red Wi-Fi convencional. Esto nos permite un entendimiento mayor de cómo *wpa_supplicant* maneja y configura las redes, y al mismo tiempo nos permite abrir un campo de posibilidades, ya que una vez desconectados del Grupo P2P podríamos intentar restablecerlo en el reinicio, o simplemente obtener de nuevo una cadena hexadecimal que compartir con otro dispositivo a partir de esta red almacenada.

5.2 Conexión a un grupo P2P complejo mediante invitación por NFC

Paso 1:

Comenzaremos por habilitar la tarjeta de red levantando el interfaz correspondiente y comprobando su estado, mediante los comandos de "ifconfig"

```
debian@debian:~$ sudo ifconfig wlan0 up
debian@debian:~$ sudo ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:b3:a7:1f
          inet addr:192.168.1.48  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:feb3:a71f/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:951 errors:0 dropped:0 overruns:0 frame:0
          TX packets:99 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:94084 (91.8 KiB)  TX bytes:11725 (11.4 KiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:38 errors:0 dropped:0 overruns:0 frame:0
          TX packets:38 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:3779 (3.6 KiB)  TX bytes:3779 (3.6 KiB)

wlan0     Link encap:Ethernet  HWaddr f4:f2:6d:0d:7c:14
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
```

Figura 42: Habilitar interfaz wlan0

Paso 2:

A continuación deshabilitamos los servicios: *wpa_supplicant*, *NetworkManager*, *DNSMasq* y *HostApd*. Tras lo que iniciaremos el *wpa_supplicant* en nuestro interfaz, indicando nuestro fichero de configuración[A].

```
debian@debian:~$ sudo wpa_supplicant -Dnl80211 -iwlan0 -cp2p.conf
Successfully initialized wpa_supplicant
```

Figura 43: Arranque *wpa_supplicant*

Al igual que en la prueba anterior queremos ver todos los eventos que pasan a través del *wpa_supplicant* en esta misma interfaz, así que lo dejamos corriendo en primer plano. Abrimos una nueva interfaz de consola para restablecer los servicios deshabilitados e inciamos la *wpa_cli*, para conectarnos automáticamente al interfaz correspondiente.

```
debian@debian:~$ sudo wpa_cli
wpa_cli v2.6
Copyright (c) 2004-2016, Jouni Malinen <j@w1.fi> and contributors

This software may be distributed under the terms of the BSD license.
See README for more details.


Selected interface 'wlan0'

Interactive mode
```

Figura 44: Inicio *wpa_cli*

Paso 3:

Empleando los comandos[B] en la interfaz abierta[44], iniciamos la búsqueda de dispositivos con capacidad Wifi-Direct. En esta prueba nuestro objetivo será establecer la comunicación con un dispositivo Smart phone. Para ello ejecutamos el comando "p2p_find"[B], haciéndonos así visibles al resto de dispositivos.

```
p2p_find
> OK
<3>CTRL-EVENT-SCAN-STARTED
<3>CTRL-EVENT-SCAN-STARTED
<3>CTRL-EVENT-SCAN-STARTED
<3>CTRL-EVENT-SCAN-STARTED
<3>CTRL-EVENT-SCAN-STARTED
<3>CTRL-EVENT-SCAN-STARTED
<3>CTRL-EVENT-SCAN-STARTED
<3>P2P-DEVICE-FOUND 92:67:1c:2e:89:51 p2p_dev_addr=92:67:1c:2e:89:51 pri_dev_type=
10-0050F204-5 name='HUAWEI G7-L01_4416' config_methods=0x188 dev_capab=0x25 group_
capab=0x0 new=1
<3>CTRL-EVENT-SCAN-STARTED
<3>CTRL-EVENT-SCAN-STARTED
<3>CTRL-EVENT-SCAN-STARTED
top
> OK
<3>P2P-FIND-STOPPED
```

Figura 45: "p2p_find" en *wpa_cli*

Podemos observar cómo se inicia el evento y encuentra un dispositivo disponible para establecer una comunicación Wifi-Direct. Como se trata del dispositivo que deseábamos, ejecutamos "p2p_stop"[B] para detener la búsqueda de más dispositivos. Como comentábamos en los primeros pasos, conviene mantener abierta la ventana con el *wpa_supplicant* en ejecución, porque nos facilita información sobre los eventos.

```
debian@debian:~$ sudo wpa_supplicant -Dnl80211 -iwlan0 -cp2p.conf
Successfully initialized wpa_supplicant
wlan0: CTRL-EVENT-REGDOM-CHANGE init=USER type=COUNTRY alpha2=ES
P2P-DEVICE-FOUND 92:67:1c:2e:89:51 p2p_dev_addr=92:67:1c:2e:89:51 pri_dev_type=
10-0050F204-5 name='HUAWEI G7-L01_4416' config_methods=0x188 dev_capab=0x25 gro
up_capab=0x0 new=1
P2P-FIND-STOPPED
```

Figura 46: "p2p_find" en *wpa_supplicant*

Como se puede comprobar, la información del evento nos aparece claramente en la ventana corriendo el *wpa_supplicant*, sin toda la ejecución de por medio lo que nos facilita bastante la tarea a la hora de entender rápidamente lo que ocurre. Únicamente falta por ver si nuestro terminal ha sido detectado por el dispositivo smart phone 'HUAWEI G7'.

Si miramos en el panel de conexión Wifi-Direct de nuestro smart phone, en el cual tenemos que estar desde el principio de la prueba, pues de otra forma el terminal Linux nunca lo detectaría y viceversa.

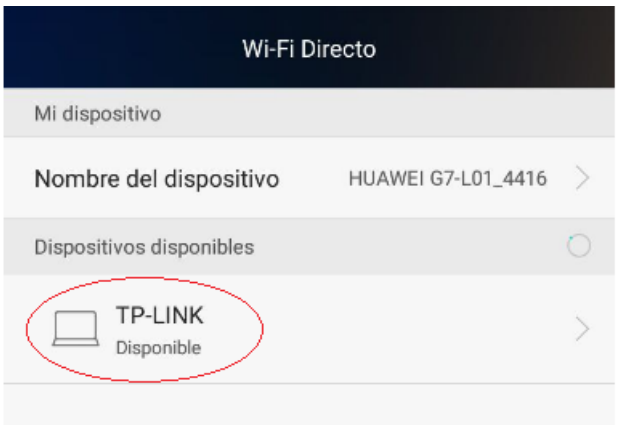

Figura 47: Wifi-Direct en Huawei

Efectivamente, se han encontrado, por tanto nos queda proceder con la conexión. Como primera medida vamos a ejecutar una comunicación entre nuestra terminal Linux y el teléfono smart phone, en la que el terminal Linux hará las veces de "group owner" y por tanto llevará la iniciativa en la conexión.

Teniendo la dirección *MAC* del dispositivo smart phone, podemos ejecutar el comando "p2p_connect"[B] y en este caso utilizar la opción "pbc" para la autenticación.

```
p2p_connect 92:67:1c:2e:89:51 pbc
> OK
<3>P2P-GO-NEG-SUCCESS role=GO freq=2462 ht40=1 peer_dev=92:67:1c:2e:89:51 peer_i
face=92:67:1c:2e:89:51 wps_method=PBC
<3>P2P-GROUP-FORMATION-SUCCESS
<3>P2P-GROUP-STARTED p2p-wlan0-0 GO ssid="DIRECT-aa" freq=2462 passphrase="gLSx0
ZJu" go_dev_addr=f4:f2:6d:0d:7c:14
<3>AP-STA-CONNECTED 92:67:1c:2e:89:51 p2p_dev_addr=92:67:1c:2e:89:51
q
```

Figura 48: Ejecutamos "p2p_connect"

Y a los pocos segundos aparecerá la invitación en nuestro dispositivo smart phone. Pulsamos conectar y esperamos a que la conexión tenga éxito.

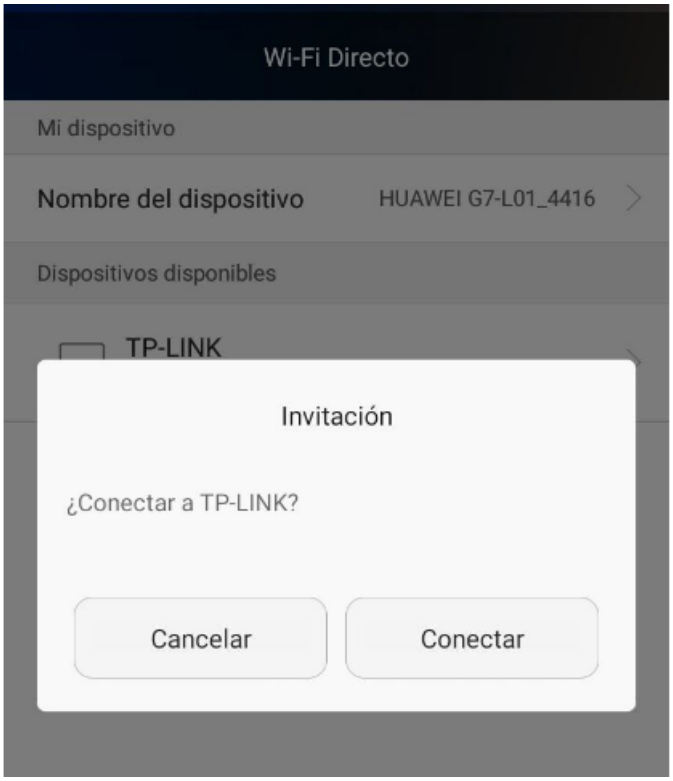

Figura 49: Invitación para la conexión Wifi-Direct mediante PBC

Como podemos ver en la figura[48], se produce una negociación en la que nuestro terminal Linux se establece como propietario del grupo, estableciendo una serie de parámetro que seguiremos viendo más adelante. Al final de la imagen se aprecia el evento que nos muestra qué dispositivo se ha conectado a este grupo, en este caso vemos que es la dirección *MAC* correspondiente con el smart phone. No obstante comprobamos que efectivamente nuestro smart phone se ha conectado.

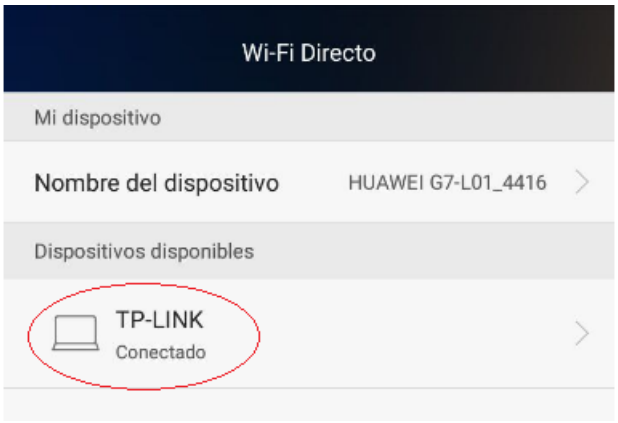

Figura 50: El dispositivo 'Huawei' está conectado.

Paso 4:

Debido a la configuración[B] establecida a la hora de iniciar el *wpa_supplicant*, se ha creado una interfaz nueva asociada al grupo creado para la conexión. Esto se puede ver en los eventos de la figura[48], junto a la información de la 'ssid' asignada al grupo y otros campos interesantes.

```
<3>P2P-GROUP-STARTED p2p-wlan0-0 GO ssid="DIRECT-aa" freq=2462 passphrase="gLSx0
ZJu" go_dev_addr=f4:f2:6d:0d:7c:14
```

Figura 51: Información asociada al grupo P2P

Por tanto en este paso nos vamos a conectar al nuevo interfaz, de la misma forma que lo hicimos al principio de la prueba. Primero comprobamos que la interfaz está correctamente levantada, ejecutando el comando "ifconfig"

```
p2p-wlan0-0 Link encap:Ethernet  HWaddr f6:f2:6d:0d:7c:14
          inet6 addr: fe80::f4f2:6dff:fe0d:7c14/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:18 errors:0 dropped:2 overruns:0 frame:0
          TX packets:43 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:3408 (3.3 KiB)  TX bytes:8129 (7.9 KiB)
```

Figura 52: Interfaz "p2p-wlan0-0"

Lo siguiente será salir del cliente *wpa_cli* simplemente escribiendo 'q', como se puede ver en la figura[48]. Ahora simplemente repetimos el paso 2 y ejecutamos *wpa_cli*, que en este caso se conectará automáticamente a nuestra interfaz deseada "p2p-wlan0-0". No obstante también podemos añadir la opción "-i" y escribir la interfaz que deseemos.

Una vez dentro nos aprovecharemos de las funcionalidades de NFC en *wpa_supplicant*, para esta primera prueba ejecutaremos el siguiente comando

"wps_nfc_config_token", que nos devolverá una cadena hexadecimal con la información del grupo P2P.

```
Selected interface 'p2p-wlan0-0'

Interactive mode

> wps_nfc_config_token NDEF
D2178F6170706C69636174696F6E2F766E642E7766612E777363100E006C1026000101104500094
49524543542D6161100300020020100F000200081027004030386335303739623130393462386362
643932373936343365656532623465626632653839353961383933353034306530623566393563626
666266356534616210200006000000000000103C000101100100020008B10200006F6F26D0D7C1410
49000600372A000120
```

Figura 53: Ejecutamos wps_nfc_config_token NDEF

Como podemos comprobar en la imagen[53], estamos en la interfaz específica del grupo y al ejecutar el comando, obtenemos una cadena de valores como esperábamos. Esta cadena puede escribirse en un TAG-NFC en formato NDEF, puesto que ya posee los valores de encapsulación necesarios. No obstante esta tarea ha de realizarse a través de aplicaciones externas.

Una vez escrita la información en un TAG-NFC, la llevaremos a nuestro segundo terminal Linux, llamado Linux-Dolly similar al primero. Realizaremos los pasos 1 y 2 de la misma manera. Recordamos que la configuración establecida en Linux-Dolly es ligeramente distinta para que no exista confusión en las capturas. Utilizaremos el interfaz wlan1, con la versión 2.x de la tarjeta de red usada en el terminal Linux. También se emplea una versión modificada del fichero de configuración[B]para el *wpa_supplicant*.

Al igual que para la escritura, la lectura del TAG-NFC se realizará mediante una aplicación externa que nos debería devolver la misma cadena de valores hexadecimales que escribimos. Una vez la tenemos, utilizaremos el comando "$wps\_nfc\_tag\_read$" en nuestro terminal Linux-Dolly, que ya estará con el *wpa_supplicant* corriendo y el *wpa_cli* listo para interactuar.

```
> wps_nfc_tag_read D2178F6170706C69636174696F6E2F766E642E7766612E777363100E006C1
026000101104500094449524543542D6161100300020020100F000200081027004030386335303739
962313039346238636264393237393634336565653262346562663265383935396138393335303430
065306235663935636266626635653461621020000600000000000103C000101100100020008B102
00006F6F26D0D7C141049000600372A000120
OK
```

Figura 54: Ejecutamos wps_nfc_tag_read en Linux-Dolly

Podemos apreciar en la imagen cómo se ejecuta el comando introduciendo la cadena hexadecimal, y esta es aceptada por nuestra terminal.

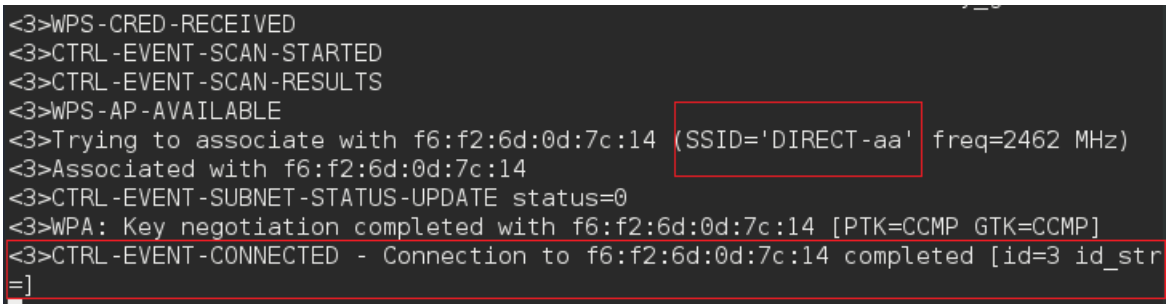

Figura 55: Evento iniciado por la información del TAG-NFC en Linux-Dolly

Vemos que la ejecución dispara una serie de eventos que empiezan por la recepción de los credenciales de WPS, lo podemos ver en el primer evento 'WPS-CRED-RECEIVED' que nos indica que nuestro comando se ha ejecutado, seguidamente inicia un proceso de escaneo en busca del punto de acceso. Enseguida vemos que se produce una comunicación e intenta asociarse al punto de acceso cuya información hemos recibido por NFC, con la dirección *MAC* del 'SSID', que veíamos en la figura[51]. Finalmente se confirma que se ha establecido la conexión.

Cuando miramos la ventana del *wpa_supplicant* corriendo, encontramos lo siguiente.

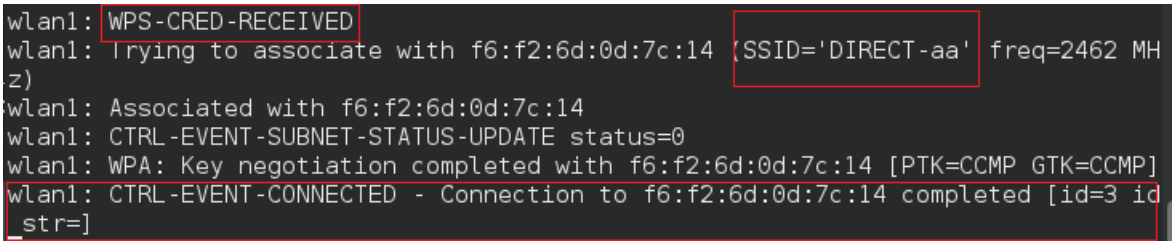

Figura 56: Caption

Podemos observar que la conexión ha sido completada y existe comunicación entre los dos terminales y el dispositivo smart phone, únicamente con la información proporcionada en el token de configuración del propio *wpa_supplicant* que podremos compartir mediante el uso de TAG-NFC. No obstante aquí no termina la prueba.

Tenemos que comprobar cómo ha afectado este proceso en el fichero de configuración de nuestro terminal Linux-Dolly.

```
network={
        ssid="DIRECT-aa"
        psk=08c5079b1094b8cbd9279643eee2b4ebf2e8959a8935040e0b5f95cbfbf5e4ab
        proto=RSN
        key_mgmt=WPA-PSK
        pairwise=CCMP
        auth_alg=OPEN
}
```

Figura 57: Fichero de configuración en Linux-Dolly

Observamos que la información de la conexión con el grupo P2P ha sido almacenada en el fichero de configuración[B] como si se tratase de una red Wi-Fi. Esto no solo nos da una visión más clara de cómo *wpa_supplicant* maneja las conexiones Wifi-Direct, sino que además nos da la certeza de que estas son almacenadas y por tanto el propio *wpa_supplicant* tratará de conectarse nuevamente en caso de que lo reiniciásemos. Esto nos permite utilizar el mismo comando que usábamos para general la cadena hexadecimal, para obtenerla nuevamente a partir de nuestro fichero de configuración y así compartir la información de la red a partir de un dispositivo que ya se ha conectado previamente.

```
> wps_nfc_config_token NDEF 0
D2175A6170706C69636174696F6E2F766E642E7766612E777363100E004C1026000101104500094
449524543542D6161100300020020100F000200081027002008C5079B1094B8CBD9279643EEE2B4
EBF2E8959A8935040E0B5F95CBFBF5E4AB1020000600000000000010490006003 72A000120
>
```

Figura 58: Ejecutamos 'wps_nfc_config_token NDEF 0' en Linux-Dolly

Como vemos, al ejecutar el mismo comando que en la figura[53] añadiendo el número de la red almacenada en el fichero de configuración, en este caso el '0' porque es la única que tenemos y por tanto está la primera, obtenemos una cadena hexadecimal con la información del grupo P2P al que estábamos conectados. Nuevamente esta información podríamos escribirla en un TAG-NFC y compartirla con otros terminales.

Paso 5:

Cabe probar una de las opciones que nos facilita el comando "wps_nfc_config_token", que hasta ahora lo habíamos empleado con la opción 'NDEF' que nos devolvía una cadena de valores con la información adicional necesaria para escribir en un TAG-NFC usando encapsulación 'NDEF'. Por tanto ahora nos falta probar qué resultado nos proporciona el mismo comando cuando ejecutamos la opción WPS. Que efectivamente nos devolverá otra cadena hexadecimal conteniendo únicamente la información para la configuración 'WPS' de la red, pero no la información adicional necesaria para la encapsulación 'NDEF'. Por supuesto esto no significa

que no pueda escribirse en un TAG-NFC. Se puede hacer en forma de archivo de texto plano. Sin embargo para esta prueba nos es indiferente, puesto que la lectura y escritura se llevan a cabo de forma externa al *wpa_supplicant*.

En cualquier caso ejecutamos nuevamente el comando en nuestra terminal Linux para obtener la cadena hexadecimal.

```
wps_nfc_config_token WPS
> 100E006C10260001011045000944495245435420616110030002002010 0F00020081027004030
38633530373962313039346238636264393237393634336565653262346562663265383935396138
39333530343065306235663935636266626635653461621020000600000000000103C0001011001
0002000B10200006F6F26D0D7C141049000600372A000120
```

Figura 59: Ejecutamos 'wps_nfc_config_token WPS'

Una vez más transportamos esta información a nuestra terminal Linux-Dolly, teniendo en cuenta que ahora la encapsulación NDEF ya no es tan sencilla. En nuestra terminal Linux-Dolly, tendremos que ejecutar "disconnect" para desconectarnos del grupo P2P y así poder comprobar qué ocurrirá cuando introduzcamos la nueva cadena hexadecimal.

```
<3>CTRL-EVENT-DISCONNECTED bssid=f6:f2:6d:0d:7c:14 reason=3 locally_generated=1
wps_nfc_tag_read 100E006C10260001011045000944495245435420616110030002002010 0F000
20008102700403038633530373962313039346238636264393237393634336565653262346562663
26538393539613839333530343065306235663935636266626635653461621020000600000000000
0103C0001011001000 2000B10200006F6F26D0D7C141049000600372A000120
> OK
<3>WPS-CRED-RECEIVED
<3>CTRL-EVENT-SCAN-STARTED
<3>CTRL-EVENT-SCAN-RESULTS
<3>WPS-AP-AVAILABLE
<3>Trying to associate with f6:f2:6d:0d:7c:14 (SSID='DIRECT-aa' freq=2462 MHz)
<3>Associated with f6:f2:6d:0d:7c:14
<3>CTRL-EVENT-SUBNET-STATUS-UPDATE status=0
<3>WPA: Key negotiation completed with f6:f2:6d:0d:7c:14 [PTK=CCMP GTK=CCMP]
<3>CTRL-EVENT-CONNECTED - Connection to f6:f2:6d:0d:7c:14 completed [id=4 id_str
=]
```

Figura 60: 'Ejecutamos 'wps_nfc_tag_read' en Linux-Dolly de nuevo

Podemos observar el orden de los eventos, donde nos hemos desconectado del grupo P2P como primera medida. Introducimos la nueva cadena hexadecimal y vemos cómo enseguida se reciben las credenciales WPS e inicia el mismo proceso que veíamos la primera vez en el caso de la figura[55]. Se ejecuta el escaneo en busca del punto de acceso y enseguida se asocia con el 'SSID' del grupo para finalmente conectarse. Esto nos hace darnos cuenta de que a efectos reales la única diferencia a la hora de tramitar un token de configuración, es la capacidad de encapsulación NDEF.

Incluso si observamos el fichero de configuración de Linux-Dolly, nos encontramos lo mismo que en la figura[57], no se ha modificado ningún parámetro de la

red.

Por supuesto también podríamos exportar la información del grupo P2P para compartirla nuevamente al igual que hacíamos hacíamos anteriormente en la figura[58], pero utilizando 'WPS' en lugar de la opción 'NDEF'. La única diferencia aparte del valor de la cadena hexadecimal, sería la capacidad de encapsulación NDEF.

6 Work summary, Conclusion and Future Work

6.1 Brief summary and Conclusion

Every day new applications are born to improve our lives, taking advantage of many different technologies that are indeed present in many aspects of ourselves. We have smart phones, smart watches, smart TVs and from a few years ago we also have IoT devices. Or in other, we live in an era were the globalisation is no longer a concept but a fact. That implies striving for better and many different ways of communication.

But it is also a reality the importance of making communication easy and accessible to everyone. Meaning the more complex technologies are brought into our everyday devices, so that developers can explore new ranges of possibilities. This could be the case of Wi-Fi Direct, a technology found in many devices at everyone's reach, and yet not recognised enough. Why is this? It could be because of lack of applications for this technology. This is true, although we have seen the many services it implements and its capabilities, there are not much user friendly applications in the market. However it is also true the existence of some applications already found at home, like Miracast[12] which we have already spoken of.

For us one of the most interesting aspects of Wi-fi Direct, was the possibility of establishing a connection between two devices without the need of a router to manage them, and with the same capabilities of a standard Wi-Fi communication. This proves significant for it changes completely the way of understanding networks. During this project we saw a few examples, mainly focused on simple networks like the ones at home or a small office, were we suggested the possibilities of interconnecting many different and incompatible devices through the use of Wi-Fi Direct, e.g. we could see a Wi-Fi Router providing Internet access to some desktop computer that is also transferring documents through a Wi-Fi Direct connection; while on the other side of the room we could have two devices sharing files and printing them, all without any intervention from the Wi-Fi Router, just using Wi-Fi Direct.

As a different perspective, we presented the NFC technology, capable of communicating two devices by approaching them very close together. We also learned of the NFC-TAGs and how do they work, being the major complement for the NFC devices and with a great variety and types, which provide us with the possibility of choosing the right NFC-TAG for our application.

Although NFC is present in many smart phones is not very popular yet. But unlike Wi-Fi Direct, it is becoming more known and used thanks to its services and applications like contact-less payment or as an access card.

NFC meant security for us. We were deeply interested in the concept of improving security through physical access. Other than its own services, NFC can be used to bootstrap and launch a more capable connection, e.g linking some Bluetooth

headphones to the smart phone just by touch; or opening a web page after touching some NFC-TAG. Here is where we found the motivation for this project, and study the possibility of bringing NFC physical security, together with the connection capabilities of Wi-Fi Direct.

We have seen that NFC is implemented in the WPS standard used by Wi-Fi Direct, where we already spoke of what possibilities we had with WPS NFC, and what tools were needed to establish a Wi-Fi Direct connection.

So taking advantage of all the things learned along this project, we run some tests. First we created a simple Wi-Fi Direct connection involving two Linux systems, where the NFC support allowed us to establish this connection. In the second test, we created a connection between a Linux system and a smart phone, then we invited the other Linux terminal with NFC.

The first set up can be seen as secure network establishment with NFC, where the Linux terminal acting as an access point, only can be reached through through the parameters written in the NFC-TAG. E.g. we could imagine an application with some data-center or simply a data-base containing critical information, therefore is not accessible from the Internet, neither can be locally accessible; we could set some specific terminal with an NFC interface in order to perform secure Wi-Fi Direct communications with such data-base, where you can only access with a specific card. Of course this is a very military oriented example, but it makes a great idea of what we are achieving.

The second set up is a perfect example for what new and modern networks can be like. There was a connection going on between two devices, and a third one was invited to the communication. Maybe we could imagine the third terminal as a device that doesn't has a display, nor a keyboard feature, so we can't input data. Therefore the only way for it to join the connection is when invited through NFC. E.g. We could have a Wi-Fi Direct connection between a smart phone and a computer where we are reviewing files from the smart-phone, and suddenly want to print one of the files; we could maybe tap the smart phone with the printer so it prints the file from the computer.

In general this are examples that prove us the amazing possibilities of the support of NFC in Wi-Fi Direct communications. Where both technologies are highly available in most of our everyday dices like smart phones. So this initial work should inspire other developers in studying and experimenting with these technologies brought together, which with the correct use of new and easy applications, could revolutionise the way we understands the networks and its services right now.

As for the project itself, we can say we met the objectives successfully, gaining a deeper knowledge and understanding of both technologies and how we can take advantage of them. Being able to show two very different scenarios of how they can be brought together. And which have proven us that yes, it is possible to add NFC support for Wi-Fi Direct communications, taking advantage of all the capabilities

of both technologies.

6.2 Future Work

Once we've seen the conclusions we have made, many lines of future work have opened to us. As a first extension of this project, we propose the develop of a tool to simplify and make the use of both technologies easier for the users. It could include a graphic interface in order to manage the set up of Wi-Fi Direct communications and the use of its services together with NFC. Being able to directly connect them just by touch or create access cards, besides including the capabilities of sharing files or other information.

Another alternative that we have barely touched, is the support of Wi-Fi Direct and NFC in Android systems, which would improve the possibility of developing apps that support both technologies.

It could be used in the IoT industry as a way of synchronising devices to interconnect them and having them centralised. You could could manage few devices with a smart phone or computer by simply touching them.

And for last, an analysis and design of a quick payment system taking advantage of NFC and Wi-Fi Direct, so that you can use contact-less payment anywhere on the store. You could identify yourself with some special NFC-TAG for clients when you go into the store in order to connect via Wi-Fi P2P with a payment server, and simply buy items by touching them with NFC.

7 Glosario

- Ad-Hoc - Tipo de red inalámbrica que no depende de una infraestructura ya existente.
- AES - Advanced Encryption Standard, consiste en un esquema de cifrado por bloques.
- AP - Access Point, punto de acceso, normalmente una estación base que conecta a los clientes de una red.
- ASP - Application Service Platform, Pataforma de servicios de aplicación, módulo lógico que necesitan los servicios en Wi-Fi Direct.
- Data Center - Centro de datos, hace referencia a centros donde se instalan todos los equipos y servidores relacionados.
- Bluetooth - tecnología inalámbrica que permite la transmisión de datos y voz entre distintos dispositivos.
- BSSID - Basic Service Set Identifier, dirección MAC del punto de acceso.
- Contactless - Sin contacto, hace referencia a las tecnologías NFC/RFID y Smartcards.
- D-Bus - sistema de intercambio de mensajes entre aplicaciones y procesos.
- DHCP - Dynamic Host Configuration Protocol, protocolo de configuración dinámica de "hosts", servidor que asigna las direcciones IP.
- DLNA - Digital Living Network Alliance, interfaz de control para el usuario que permite la reproducción de medios multimedia a través de Wi-Fi Direct.
- DNS - Domain Name Server, sistema de nomenclatura jerárquico para dispositivos de redes IP.
- *Dnsmasq* - servicio capaz de actuar como servidor DHCP o proporciona caché DNS.
- EAP - Extensible Authentication protocol, entorno con varios métodos de autenticación.
- EAPOL - EAP Over Lan, protocolo usado en LAN para transportar EAP.
- GO - Group Owner, propietario de grupo, se trata de uno de los roles que se pueden ejecutar un grupo P2P.
- *HostApd* - Herramienta que permite configurar un AP.
- HTTP - Hypertext Transfer Protocol, protocolo de transferancia de hipertexto, permite la transmisión de información en internet.
- IoT - Internet of Things, internet de las cosas, hace referencia la interconexión digital de objetos cotidianos con internet.

- IP - Internet Protocol, protocolo de internet.
- IPP - Internet Printer Protocolo, protocolo de internet para la impresión.
- Kernel - Software que constituye una parte fundamental del sistema operativo.
- LAN - Local Area Network, red de área local.
- Linux - Denomina de forma coloquial el conjunto de sistemas operativos con kernel Linux.
- LLC - Logical Linc Control, control de enlace lógico, hace referencia al enlace establecido en LLCP.
- LLCP - Logical Link Control Protocol, protocolo de control de enlace lógico, usado por NFC para realizar comunicaciones Peer-to-Peer.
- LLP - Logical Link Protocol, hace referencia a la comunicación mediante LLCP.
- MAC - también conocido como dirección física, identifica de forma única un dispositivo de red.
- NDEF - NFC Data Exchange Format, formato de intercambio de datos por NFC.
- NFC - Near Field Communication, comunicación de campo cercano.
- P2P - Peer to Peer, comunicación o red entre iguales.
- PDU - Payload Data Unit, unidad de datos de contenido, usado en LLCP.
- PBC - Push Button Control, autenticación mediante pulsación de botón en patalla.
- PC - Personal Computer, ordenadore personal.
- PIN - Personal Identificatioin Number, contraseña o clave numérica.
- PSK - Pre-Shared Key, clave pre-compartida, utilizada en cifrado WEP y WPA.
- RFID - Radio Frequency IDentification, identificación por radio frecuencia, tecnología anterior a NFC que permitía la transmisión de datos a elementos pasivos a través de radio frecuencia.
- RSN - Robust Security Network, implementación de Wi-Fi Alliance de la norma 802.11i como WPA2, que emplea encriptación AES.
- Smartcard - Tarjeta inteligente usada en RFID/NFC, con capacidades de almacenamiento.

- Smart Phone - Teléfono inteligente, normalmente con capacidades Wi-Fi o más.
- Smart TV - Televisión inteligente, normalmente con capacidades Wi-Fi o más.
- TAG-NFC - Dispositivos pasivos con capacidades NFC, normalmente en forma de tarjeta.
- TAG-RFID - Dispositivo pasivo con capacidades RFID, normalmente en forma de etiqueta.
- TKIP - Temporal Key Integrity Protocol, protocolo de integridad temporal de clave, usado en WPA.
- UHF - Ultra High Frequency, frecuencia ultra alta, es una banda de frecuencias en el rango de 300 MHz a 3 GHz.
- USB - Universal Serial Bus, de forma coloquial hace referencia a las memorias flash con esta conexión.
- VirtualBox - Software de virtualización de sistemas operativos.
- WEP - Wired Equivalent Privacy, protocolo de encriptación por defecto para redes 802.11.
- WDFS - Wi-Fi Direct Services, servicios de Wi-Fi Direct, incluye librerías para la implementación de dichos servicios.
- Wi-Fi - Wireless Fidelity, estándar de comunicación inalámbrica basado en 802.11.
- Wi-Fi Direct - también WiFi-Direct y WiFi P2P, tecnología de la Wi-Fi Alliance que permite las conexiones directas usando Wi-Fi.
- WLAN - Wireless Area Network, conexión inalámbrica en área.
- WPA - Wireless Protected Access, implementación del estándar 802.11i basada en TKIP
- *Wpa_supplicant* - proceso "demonio" que controla las conexiones inalámbricas y la seguridad.
- *Wpa_cli* - Interfaz que permite controlar el *wpa_supplicant* mediante comandos de texto.
- *Wpa_gui* - Interfaz gráfica que permite controlar el *wpa_supplicant*
- WPS - Wi-Fi Protected Setup, estándar de Wi-Fi Alliance para facilitar la creación de redes WLAN.

8 Referencias

Referencias

[1] Wi-Fi Peer-to-Peer(P2P) Technical Specification, *Wi-Fi Alliance*, ver. 1.7, 2016.

[2] Device to device communications with WiFi Direct: overview and experimentation, *Daniel Camps-Mur, Andres Garcia-Saavedra and Pablo Serrano*, 2013.

[3] Análisis de Vulnerabilidades de Redes Inalámbricas Wifi-Direct, *Amador Díaz Gervasio*, 2016.

[4] A Survey Paper on Radio Frequency IDentification (RFID) trends, *Christoph Jechlitschek*, pp.2-9, 2013.

[5] RFID: Technology and Applications, *Sridhar Iyer*, pp.6-14,18-24,26-31, 2005.

[6] ISO/NFC Standards and Specifications Overview, *Texas Instruments*, NFC/RFID Training Module#1, pp.52-56, pp.59, 2014.

[7] UPnP File Transfer Service Technical Specification, *Wi-Fi Alliance*, ver. 1.1, pp.6-9, pp.20, 2015

[8] Wi-Fi Peer-to-Peer Services Print (P2Ps-Print) Technical Specification (for Wi-Fi Direct® services certification), *Wi-Fi Alliance*, ver. 1.1, pp.12-18, 2014.

[9] Wi-Fi Peer-to-Peer Services (P2Ps) Technical Specification (for Wi-Fi Direct® services certification), *Wi-fi Alliance*, ver. 1.2, 2015.

[10] WFDS-A new Emerging technology over Wi-Fi Direct, *Ranjani H R, Dr.Radhika K R, Vinoth Sampath*, 2014.

[11] Wi-Fi Display Technical Specification, *Wi-Fi Alliance*, ver. 2.1,pp.20-27, 2017.

[12] Wi-Fi CERTIFIED Miracast™ Technical Overview, *Wi-Fi Alliance*, 2017.

[13] Wi-Fi Protected Setup Specification, *Wi-Fi Alliance*, v1.0h, 2006.

[14] Wi-Fi CERTIFIED Wi-Fi Protected Setup™: Easing the User Experience for Home and Small Office Wi-Fi® Networks, *Wi-Fi Alliance*, updated ver. 2014.

[15] Wi-Fi CERTIFIED Wi-Fi Protected Setup™: Easing the User Experience for Home and Small Office Wi-Fi® Networks, *Wi-Fi Alliance*, ver. 2.0.1, 2010.

[16] Wi-Fi CERTIFIED Wi-Fi Protected Setup™ adds NFC "tap-to-connect" for simple set up of security-protected Wi-Fi® devices and networks, *Chris Nishimura*, 2014, *https://www.wi-fi.org/news-events/newsroom/wi-fi-certified-wi-fi-protected-setup-adds-nfc-tap-to-connect-for-simple-set-up.*

[17] Wi-Fi Alliance® pushes the NFC button, *Frank Dwidowsky*, 2014, *https://www.wi-fi.org/beacon/frank-dawidowsky/wi-fi-alliance-pushes-the-nfc-button.*

[18] WPA Supplicant, *Jouni Malinen*, 2003, *https://w1.fi/cgit/hostap/plain/wpa_supplicant/README.*

[19] wpa_supplicant and Wi-Fi P2P, *Jouni Malinen*, *https://w1.fi/cgit/hostap/plain/wpa_supplicant/README-P2P.*

[20] Example wpa_supplicant configuration file, *Jouni Malinen*, 2003, *https://w1.fi/cgit/hostap/plain/wpa_supplicant/wpa_supplicant.conf.*

[21] wpa_supplicant and Wi-Fi Protected Setup (WPS), *Jouni Malinen*, 2003, *https://w1.fi/cgit/hostap/plain/wpa_supplicant/README-WPS.*

[22] Nfc Communicators and Nfc Communications Enabled Devices, *Peter Robert Symons*, patent US20080272889 A1, pp. 1-9, 2008.

[23] Specifications and Application Documents, *Laurent Sourgen(NFC Forum Board Member), STMicroelectronics*, 2012.

[24] Specifications and applications documents, *NFC Forum*, 2017 *https://nfc-forum.org/our-work/specifications-and-application-documents/specifications/nfc-forum-technical-specifications/#data.*

[25] Logical Link Control Protocol Technical Specification, *NFC Forum*, ver. LLCP 1.0, 2009.

[26] NFC Data Exchange Format (NDEF), *NFC Forum*, ver. NDEF 1.0, 2006.

[27] FeliCA Card User's manual Excerpted Edition, *Sony*, ver. 2.02, pp.6-8, pp.47-48, pp.54-59, pp.93, 2017.

[28] MiFare Standard Card IC MF1 IC S50 Functional Specification, *Philips*, ver. 4.0, pp.3-15, 1998.

[29] Chip Card ICs, *NXP Semiconductors Austria GmbH Styria*, *https://www.mifare.net/en/products/chip-card-ics/*, 2017.

[30] NFC Tools, nfc-tools developers community, *http://nfc-tools.org/index.php?title=Main_Page*, 2013.

[31] Beginning NFC: Near Field Communication with Arduino, Android, and PhoneGap, *Tom Igoe,Don Coleman,Brian Jepson*, pp.203-208, 2014.

[32] COIT. Verificación de proyectos de ICT, COIT, *http://www.coit.es/web/ejercicio profesional/svc/dv/2016 vericacion ict.pdf*

Anexos

A Anexo I: Organización del proyecto y Presupuesto

A.1 Organización del proyecto

A continuación vamos a mostrar una distribución de las horas y organización del proyecto. Teniendo en cuenta que este proyecto consiste en un Trabajo de Fin de Grado, se requieren un total de 300 horas que se reparte de la siguiente manera.

- **Documentación:** Investigación de las tecnologías expuestas en el trabajo durante las primeras fases del mismo. Durante las fases de pruebas, estudio del funcionamiento y manejo de las herramientas necesarias para utilizar las tecnologías.
- **Pruebas:** Esta parte supone la de mayor tiempo de todas y depende enormemente de la fase de estudio. Implica la utilización de la tecnología en cuestión, e ir comprobando las posibilidades de combinarlas mediante el uso de herramientas específicas.
- **Resultados:** Separamos el proceso de obtención de resultados del de las pruebas, ya que una vez obtenidos los resultados hay que documentarlos y obtener conclusiones siempre y cuando resulten válidos. Esto implica repetir pruebas y tomar los datos, entre otras cosas.
- **Memoria:** La escritura de este documento requiere conocimientos de código en LaTeX. Comprobar la correcta escritura en busca de fallos ortográficos o gramaticales. Aquí incluiremos la preparación de la defensa del trabajo.
- **Tutelaje:** Esta parte es la que menos tiempo consume, incluye las horas de tutoría en despacho, consultas y correos.

Tarea		Duración
Documentación		
	Documentación sobre WiFi Direct	20 horas
	Documentación sobre NFC	12 horas
	Comprensión de *wpa_supplicant*	10 horas
	Entendimiento de 'nfc-tools' y 'libfreefare'	5 horas
Pruebas		
	Instalación de herramientas y drivers	10 horas
	Realización de pruebas	100 horas
Resultados		
	Comprobación de las pruebas con éxito	30 horas
	Toma de datos y conclusiones	10 horas
Memoria		
	Aprendizaje de LaTeX	10 horas
	Escritura del documento	73 horas
	Edición y Corrección	12 horas
Tutelaje		8 horas
TOTAL		300 horas

Cuadro 1: Organización del proyecto

A.2 Presupuesto del proyecto

Dado que para la realización del proyecto ha sido necesaria la compra de material, podemos distinguir entre costes materiales y de personal.

- **Costes de Personal:** Registran el valor monetario de las personas envueltas en el proyecto. Según el Colegio Oficial de Ingenieros de Telecomunicación[32], el salario medio rondaba los 60€en el año 2016. Podemos asumir que esta cantidad no ha variado en el año 2017, por lo que suponemos más del doble para un jefe o supervisor, supongamos 150€la hora. Y considerando que el proyecto consta de una duración de 300 horas, que se han repartido en 2 meses (5h al día).

COSTES DE PERSONAL	Coste[€/hora]	Coste Mensual[€]
Salario medio de ingeniero	60.00	9000.00
Supervisor de proyecto	150.00	600.00
Sub-TOTAL	210.00	9600.00
Coste TOTAL en 2 meses		19200.00€

Cuadro 2: Costes de Personal

- **Costes de Material:** Registran el dinero empleado en material, físico o intelectual, necesario para llevar a cabo el proyecto. En nuestro caso al utilizar herramientas de software libre, solo tendremos que incluir los gastos que supone el equipo físico.

COSTES DE MATERIAL	Coste[€/unidad]	Coste Mensual[€/unidad]
Ordenador	1200.00	100.00
Móvil Huawei G7	200.00	8.33
(2x)Antena TP-LINK	(2x)10.00	(2x)5.00
Lector NFC SCL3711	39.00	19.5
(2x)Tarjetas MIFARE Ultralight	(2x)2.00	(2x)1.00
Tarjeta FeliCa Lite	10.00	5.00
Otros Costes	50.00	25.00
Sub-TOTAL	1523	169.88
Coste TOTAL en 2 meses		339.76€

Cuadro 3: Costes en Material

- **Costes Totales:** El presupuesto total necesario para cubrir los gastos del proyecto, más los impuestos del 21 %.

COSTE TOTAL	Coste[€]	21 % IVA
Costes de Personal	19200.00	4032.00
Costes de Material	339.76	71.35
Sub-TOTAL	19539.76	4103.35
Coste TOTAL con IVA		23643.11€

Cuadro 4: Presupuesto Total del Proyecto

B Anexo II: Manual de configuración y Comandos

B.1 Configuración y arranque de *wpa_supplicant*

Con el fin de poder replicar las pruebas realizadas en la sección[5], dejamos un breve manual con los pasos y comandos utilizados.

Ficheros de configuración de las terminales Linux:

- **Configuración del terminal Linux con wlan0**.

```
ctrl_interface=/var/run/wpa_supplicant
driver_param=use_p2p_group_interface=1
update_config=1
device_name=TP-LINK
device_type=1-0050F204-1
config_methods= nfc_interface int_nfc_token virtual_display
p2p_go_intent=15
p2p_go_ht40=1
country=ES
disassoc_low_ack=1
p2p_go_max_inactivity=60
```

Figura 61: Captura de la configuración del terminal Linux

- **Configuración del terminal Linux-Dolly con wlan1**

```
ctrl_interface=/var/run/wpa_supplicant
driver_param=use_p2p_group_interface=0
update_config=1
device_name=TP-LINK-DOLLY
device_type=1-0050F204-1
config_methods= nfc_interface int_nfc_token virtual_display
p2p_go_intent=15
p2p_go_ht40=1
country=ES
disassoc_low_ack=1
p2p_go_max_inactivity=60
```

Figura 62: Captura de la configuración del terminal Linux-Dolly

Arrancando *wpa_supplicant:*

A continuación mostraremos los pasos a seguir para ejecutar *wpa_supplicant* y*wpa_cli*. Todos los comandos deben ejecutarse bajo permisos de administrador.

1. Activamos la tarjeta de red levantando el interfaz. Para ello ejecutamos 'iwconfig' para ver la interfaz donde se encuentra y luego 'ifconfig <Interfaz> up'. Podemos comprobar que se ha levantado correctamente usando 'ifconfig'.

2. Debemos detener ciertos servicios antes de inciar el *wpa_supplicant*. Introducimos los siguientes comandos:
   - 'killall wpa_supplicant'
   - 'killall dnsmasq'
   - 'service network-manager stop'
   - 'service hostapd stop'
3. Iniciamos el *wpa_supplicant* ejecutando la siguiente orden: 'wpa_supplicant -u -Dnl80211 -c p2p.conf -i wlan0'. Donde p2p.conf será el nombre de nuestro archivo de configuración y wlan0 la interfaz en la que estamos levantando el *wpa_supplicant*.
4. Abrimos una nueva ventana de terminal e introducimos: 'service hostapd stard' para reactivar el servicio.
5. Iniciamos *wpa_cli*, ejecutando el comando 'wpa_cli'.

B.2 Casos de estudio

A continuación dejamos unas tablas con los comandos ejecutados paso a paso en *wpa_cli* durante las pruebas.

Tabla resumen de una conexión WiFi-Direct usando NFC

Pasos	Terminal Linux wlan1	Terminal Linux wlan0
Paso 1	Habilitar tarjeta de red	Habilitar tarjeta de red
Paso 2	Arrancar *wpa_supplicant* y *wpa_cli*	Arrancar *wpa_supplicant* y *wpa_cli*
Paso 3	p2p_group_add	
Paso 4	wps_nfc_config_token NDEF & escribir TAG-NFC	
Paso 5		Leer TAG-NFC & wps_nfc_tag_read <Hexdump>
Paso 6	Dirección IP	Dirección IP

Cuadro 5: Resumen comandos Prueba 1

Tabla resumen de una conexión WiFi-Direct más compleja usando NFC

Pasos	Terminal Linux	Terminal Android	Terminal Linux-Dolly
Paso 1	Habilitar tarjeta de red	Activar WiFi-Direct Android	Habilitar tarjeta de red
Paso 2	Arrancar *wpa_supplicant* y *wpa_cli*		Arrancar *wpa_supplicant* y *wpa_cli*
Paso 3	p2p_find	Detección dispositivos P2P	
Paso 4	p2p_connect pbc		
Paso 5		Aceptar invitación	
Paso 6	Dirección IP	Dirección IP	
Paso 7	wps_nfc_config_token NDEF & escribir TAG-NFC		
Paso 8			Leer TAG-NFC & wps_nfc_tag_read <Hexdump>
Paso 9			Dirección IP

Cuadro 6: Resumen comandos Prueba 2

B.3 Comandos adicionales

A continuación se dejan dos tablas con comandos para realizar funciones de lectura y escritura en NFC.

Tabla resumen de comandos básicos para libnfc

Modelo	Detección	Lectura	Escritura
	nfc-list		
MiFare Classic		nfc-mfclassic r b dump.mfd	nfc-mfclassic w b dump.mfd
MiFare Ultralight		nfc-mfultralight r dump.mfd	nfc-ultralight w dump.mfd

Cuadro 7: Resumen comandos básicos libnfc

Tabla resumen de comandos básicos para libfreefare

Modelo	Detección	Lectura	Escritura
	ntag-detect		
MiFare Classic		mifare-classic-read-ndef -o ndef.bin	mifare-classic-write-ndef -i ndef.bin
MiFare Desfire		mifare-desfire-read-ndef -o ndef.bin	mifare-desfire-write-ndef -i ndef.bin

Cuadro 8: Resumen comandos básicos libfreefare

C Anexo III: Summary

C.1 Introduction

NFC technology presents a great opportunity to interact and establish communications between devices, by simply approaching them close together. This connections are pretty resource-full, for we can exchange data almost instantly, with a simple touch. We could exchange user information, small media files, perform a payment or launch a more complex and capable connection like Bluetooth. Nowadays most smart phones come with NFC integrated, allowing developers to expand our ways of using this technology; we can see NFC from social networking to secure identification, contact-less payment and even in the gaming industry. For us the main aspects of NFC technology, will be the security it provides due to its way of communicating, and the possibility of establishing a more capable and complex connection.

There are many capable and interesting wireless connections we could use to transmit data or exchange files; we could start a file transfer via Bluetooth after reading some NFC-TAG, or simply pair our headset upon touch, the same idea could be applied to set up a Wi-Fi Network.

Wi-Fi Direct is a technology that allow us to connect different devices without any router intervention, but with the capabilities of Wi-Fi connectivity. Thus bringing flexibility to the traditional way of communication, being able to establish a connection between two devices (peer to peer) or more (group P2P). WiFi-Direct can be found in many devices like smart phones, smart TVs, tablets or printers. In Linux systems we can manage WiFi-Direct connections thanks to *wpa_supplicant*, a 'daemon' process in charge of network connectivity and security access, being able to support WEP, WPA/WPA2 and also WPS. *Wpa_supplicant* will help us create and control our Wi-Fi Direct connections with a command-line tool called *wpa_cli*.

Regarding the NFC, we will use a NFC-Reader device and some NFC-Tags in order to study this technology, along with son NFC-TAGs. Using the tools freely provided by the 'nfc-tools' organisation, we will be able to perform simple read/write operations with NFC-TAGs.

With this project we intend to bring together two very powerful technologies that have been present in our daily basis for a few years now, but still are not very known even though we have it in our pockets. We seek to break this barrier opening a wide range of opportunities, so that in the future we can enjoy Wi-Fi Direct and NFC in a unique manner.

C.2 Estate of the Art

We have always understood that the purpose of technology was to make our lives easier. In this way, the human being has been able to endure and survive, getting to the point of accommodating and becoming able to perform tasks we couldn't

even think of. As for today, we see technology not only as something to make our lives easier, but also time-saving. With this thought are born many technologies, like the Internet, which brought us an infinite world of interconnection, data transferring, file sharing, and in general communicating. But in order to perform this kind of communication, we need and access point that help us connect to the Internet, this device we speak of is a router, a key component in every network.

The problem begins when we want to communicate with a device in the same network that we are. Of course we can establish a connection and exchange data, but for this task we are dependant of the router. So what we need is a technology capable of interconnecting two devices without any router intervention.

Bluetooth is a technology that allow us to establish a connection between two devices, in order to share files or transfer voice and music. This could be a solution for our problem, but still not the one we are looking for, because is not secure enough and doesn't fulfil my specifications. Maybe if there were some technology with the same capabilities of Wi-Fi, and able to perform connections without the need of a router. It exists and it's called Wi-Fi Direct. Developed by the Wi-Fi Alliance, Wifi-Direct is a technology able to connect two or more devices taking advantage of the Wi-Fi capabilities, reaching transmission rates of 250 Mbps and distances over 200m. This technology will make a solution for our problem, because it is not router-dependant and can brought us the Wi-Fi security systems among other things. Moreover Wifi-Direct can be found in many devices like Smart phones, tablets, Smart TVs and printers.

Now with WiFi-Direct implemented in our networks, we have opened a world of possibilities and interconnection: we could have a PC connected to the internet via ethernet and a printer with Wi-Fi, so that a smart phone could connect to the printer and start printing some file using Wifi-Direct, while at the same time is joining a P2P group formed between another PC and a smart TV.

However we are still thinking in a digital version of communicating, were we are implying there is some distance between the two devices. If we think of a human example of sharing, we could imagine two people handing a pencil to each other, or borrowing a book. If they were a few meters apart from each other, of course they could throw the pencil or the book. But what happens if a third person wants to steal the object, in this case both friends will simply get close together and hand the object, so that the third person can't get it easily.

The same concept applies with the NFC technology, in which communications are established upon bringing both devices close to each other (about 4-10cm). This brings us not only the security we have spoken of, but the means of understanding other ways of communication. NFC was developed by Nokia, Philips and Sony in the year 2004 with the foundation of the NFC Forum. NFC is based in RFID, and works in the frequency band of 13.56 MHz. It counts with a great complement known as NFC-TAG, an electronic device with NFC capabilities, in which you can perform read/write operations. One of the uses NFC can offer us, is the

possibility to establish a more complex communication after the NFC connection.

NFC presents us with a second solution to our problem, so why not try to put together both solutions. We could obtain the security of NFC with the capabilities of Wifi-Direct connection. In which case we would change the way we see networks today, expanding the possibilities we already have when integrating Wi-Fi Direct in our network. But now with NFC and Wifi-Direct, we could establish a Wi-Fi Direct connection with a device with a touch, or store some P2P Group parameters in a NFC-TAG so you can share that connection with other devices, we could even connect to some PC using a NFC-TAG like an access card.

C.2.1 Wi-Fi Direct in Linux

In Linux systems we will find Wi-Fi Direct thanks to the hand of the *wpa_supplicant*, although other applications are involved, this one is the main pilar of Wi-Fi Direct in Linux. *Wpa_supplicant* is a software build as a WPA Supplicant, which manages the secure establishing of connections, supporting WEP,WPA and WPA2 encryption keys.

Wpa_cli is a tool with which we can communicate with the *wpa_supplicant*, through commands. We will take advantage of this tool in order to establish Wi-Fi Direct connections and manage them. But for the connection to work, we also need the help of 'hostapd', capable of configuring our system as an access point, and work as a DHCP server. We must menton D-Bus, a message system to communicate processes and applications between them.

With all this tools we will be ready to create a Wifi-Direct connection with any other capable device. And we are set to try and study the WPS capabilities of *wpa_supplicant*, which supports NFC as a WPS mode, WPS_NFC.

C.2.2 NFC in Linux

When talking about NFC in Linux, we find the problem that usually each NFC device has its own software made by the manufacturer. Nfc-tools is an organisation founded in order to provide open software tools to manage NFC devices.

'Libnfc' is the first open software library made for NFC devices. It provides the tools to perform low-level and very simple operations like read or write, it also comes with a programmers API. This is a great advantage because of the difficulty that represent programming low-level applications with this kind of devices. 'Libfreefare' is another library build on top of 'libnfc' that implement high-level tools, including read/write operations in few MIFARE models. Finally we must mention 'nfcpy', an alternative set of Python modules providing tools for the NFC technology.

C.3 Wifi-Direct

Initially known as Wi-Fi P2P, it is a Wi-Fi standard withe the purpose of connecting devices without the need of an access point. Wi-Fi Direct works in the 802.11a/g/n frequency bands, which allow devices to use Wi-Fi and Wi-Fi Direct at the same time. Although they usually work in the 2.4 GHz frequency.

C.3.1 Architecture

Wifi-Direct communications are perform using the concept of group, known as 'P2P Group'. After some negotiating the group is established, where one of the devices acts like the AP, being called 'Group Owner', whereas the rest of the members are known as 'P2P Client'. The P2P communication can be simplified as follows:

- **Device Discovering:** Both devices start searching other Wi-Fi Direct devices, while at the same time announcing themselves. This process is performed in the 'Social Channels' which are the 1,6 and 11 of the 2.4 GHz frequency.
- **Service Discovering:** Once the devices have found each other, there's a second discovering process at a higher level, that is optional. This process will look for services in the other devices, using the 'Applications Service Platform'(ASP) in charge of finding the available services and starting them. In order to do this, an ASP session is established to negotiate the services, once both agree and the P2P connection is successful, the ASP module will wake up the service. Once the service is terminated, the ASP will close the session.
- **Group Formation:** If after finding each other the devices want to connect, the group formation process starts. First the have to exchange credentials and some basic P2P attributes, for which they can use three WPS methods: PIN, PBC, or NFC. This phase is also known as 'provisioning'. Once the credentials have been exchange, the enter into the 'GO Negotiation' phase, where the will actually negotiate upon the value of the attribute 'Go Intent', in order to see who gets to be 'Group Owner' and who 'P2P Client'. Depending on the first part, P2P groups can be considered Standard, Autonomous or Persistent.

C.3.2 WPS

WiFi-Direct implements WPS for its security. WPS is not a security mechanism for itself, but definition of many protocols like WPA/WPA2, AES-CCMP and PSK among others, with the objective of making the creation of WLANs easy. WPS follows a two phase architecture: in the first phase when an *Enrollee* wants to join the network, it has to negotiate with the *Registrar*, where they will exchange credentials and disconnect; in the second phase, the *Enrolle* will reconnect to the *Registrar* using the new credentials and accessing the network. WPS implements the modes PIN, PBC and NFC for the credentials exchange, but we can also find

a P2P invitation where the devices already know each other stepping directly into phase 2.

C.3.3 WiFi-Direct Services

The WiFi-Direct services are defined in the WFDS framework, taking advantage of ASP but also allowing it to have more than one ASP session. We can find four services implemented within WFDS.

- **Send Services:** Allow the transferring of data blocks using HTTP.
- **Play Services:** It allows the user to play media content in other devices, using a DLNA connection with a control interface.
- **Print Services:** This services is based on IPP, and brings us the capability of printing using Wi-Fi Direct connectivity.
- **Display Services**: taking advantage of all the WFDS capabilities, this services allow us to communicate two non compatible devices, being able to watch its screen content.

C.3.4 NFC in Wifi-Direct

Within the Wi-Fi Direct standard we can find support of NFC for P2P, this is because of the WPS modes in which we find NFC. The standard explains that for P2P, NFC can be used in order to discover other devices without starting the discovering step, simply by touching. This process is called 'handover' and when this happens they exchange the WPS credentials, stepping directly into the phase 2 of the WPS process. The handover can be 'negotiated' if both devices start a negotiation, or 'static' if one of them invites the other to join a connection.

C.3.5 Wi-Fi Direct and *wpa_supplicant*

We will use *wpa_supplicant* to create and manage Wi-Fi Direct Connections. For this purpose we will use the *wpa_cli*. Some of the most common commands needed for this P2P connections are: 'p2p_find', 'p2p_connect', 'p2p_group_add', and more. We will also find specific commands for WPS operations like 'wps_pbc' y 'wps_pin'.

C.4 NFC

NFC is a technology based in RFID, that allows the communication between two active devices or with a passive device. NFC Forum is an organisation founded by Sony and Philips in the year 2002. Nowadays the NFC Forum has more than 170 associates, and has made possible a growth in the NFC technology. Some of the most notable uses we can find today are: contact-less payment, SIM NFC. security access cards, electronic paper money.

C.4.1 NFC Fundamentals

NFC works using inductive coupling. Two small loop antennas create what is called 'near field' at 13.56 MHz, allowing the magnetic field to couple with another NFC device, being able to modulate the signal to establish a communication. This 'near field' has a reach of no more than 10 cm. The NFC-TAG are passive electronic devices without an energy source, they are powered by the inductive coupling. Therefore in NFC communications there will be an 'Initiator' device and a 'Target'. The NFC-TAGs are a very versatile complement of NFC, it has a chip with enough capacity to store some information. There exists four types of NFC-TAGs, each named by its number and with different capabilities.

C.4.2 NFC Devices Architecture

NFC implements three services.

- **Peer-to-Peer:** This service allows the exchange of information between devices.
- **Read/Write:** With this mode, a user can read or write in NFC-TAGs, or in a device emulating a TAG.
- **Card Emulation:** In this mode the device works as a NFC-TAG or a 'Smartcard'.

All these services can only be possible thanks to de LLCP protocol and the NDEF format.

C.4.3 LLCP

The 'Logical Link Control Protocol' provides the means to transfer information between NFC devices. The LLC module uses the PDU packet format that is able to perform the same tasks requested in the link layer. The LLC module is responsible of establishing and finishing a proper connection between the two devices. This connection can be oriented or not. The not oriented connection, also called 'connectionless transport' in the standard, consists in an 'unacknowledged' transmission system, with the less complexity possible. The connection oriented transport implements 'data link connections' and needs an acknowledge message for every packet, which makes this transport method completely reliable but time consuming.

C.4.4 NDEF

'NFC Data Exchange Format' is the standardised way of information exchange implemented by NFC. Structured in binary messages, NDEF encapsulates the content into a unique message. The NDEF messages are conformed by several 'NDEF Records' containing the information in the payload. This makes encapsulation and chunking data easier. NDEF represents the communication language of NFC, being used in all services and allowing devices to share files and understand them.

C.4.5 MiFare and FeliCa

These two are the technologies present in the NFC-TAGs.

- **FeliCa**: Designed as a 'Contact-less Smartcard' for RFID technology, but compatible with NFC, working at speeds from 212 kbps to 424 kbps. They implement DES/AES of 128b its depending of the model. They are very popular in Japan as payment cards and for the public transport.
- **MiFare:** implementing AES, DES/Triple-DES, they present a variety of models, each designed for an specific purpose. We can find the following basic models: Mifare Classic, the most basic version; Mifare Plus, considered a replacement of the classic model; Mifare Ultralight, very simple and inexpensive; MiFare SmartMX, oriented to security tasks; Mifare DESfire, a similar alternative to 'SmartMX'. MiFare cards rates go from 106 kbps to 424 kbps.

C.4.6 NFC and 'nfc-tools'

'Nfc-tools' is an organisation that provides information and open software for Linux systems. Among its tools we can find the two fundamentals for the use of NFC in Linux.

- **libnfc:** the first open library, that provides an API developing low-level applications that allow us to perform raw-bite simple NFC tasks, like reading or writing.
- **libfreefare:** build over 'libnf', 'libfreefare' offers high-level applications and an API, supporting the different types of MiFare NFC-Tags. With these tools we can perform the simple reading/writing tasks in an easier way.

C.5 Tests

We are going to try two tests in order to check the possibilities of adding NFC support to Wi-FI Direct connections. For that purpose we are creating two different set ups in which will take advantage of *wpa_supplicant* to establish and manage the connections. In the first test, we will create an autonomous P2P Group and generate a simple P2P connection using NFC; in the second test we will create a more complex environment, connecting a Linux terminal with the mobile phone and sharing this connection with another Linux terminal over NFC.

C.6 Conclusions

We have proven the possibilities of NFC support for Wifi-Direct connectivity, which should encourage future developers to try and experiment with this technologies together, opening the doors to a new era of communications and revolutionising the way we see and understand Networks today,